

Analysis of network flows in complex networks

Vom Fachbereich Informatik der Technischen Universität
Kaiserslautern zur Verleihung des akademischen Grades Doktor der
Naturwissenschaften (Dr. rer. nat.) genehmigte Dissertation von

von

Mareike Bockholt

Datum der wissenschaftlichen Aussprache: 19. Mai 2021

Dekan:

Prof. Dr. Jens Schmitt

Berichterstatter:

Prof. Dr. Katharina A. Zweig,

Prof. Dr. Ulrik Brandes

D 386

Mareike Bockholt:
Analysis of network flows in complex networks
Dissertation
Contact: mareike.bockholt@gmail.com
Supervisor: Prof. Dr. Katharina A. Zweig
Algorithm Accountabiliy Lab, Department of Computer Science,
University of Kaiserslautern, Germany

Abstract

In recent decades, there has been increasing interest in analyzing the behavior of complex systems. A popular approach for analyzing such systems is a network analytic approach where the system is represented by a graph structure [WF94, BLM⁺06, BE05, Ves18]: Nodes represent the system's entities, edges their interactions. A large toolbox of network analytic methods, such as measures for structural properties [New10], centrality measures [KLP⁺05], or methods for identifying communities [For10], is readily available to be applied on any network structure. However, it is often overlooked that a network representation of a system and the (technically applicable) methods contain assumptions that need to be met; otherwise, the results are not interpretable or even misleading. The most important assumption of a network representation is the presence of indirect effects: If A has an impact on B, and B has an impact on C, then A has an impact on C [Zwe16, BRMW13]. The presence of indirect effects can be explained by "something" flowing through the network by moving from node to node. Such network flows (or network processes) may be the propagation of information in social networks, the spread of infections, or entities using the network as infrastructure, such as in transportation networks. Also several network measures, particularly most centrality measures, assume the presence of such a network process, but additionally assume specific properties of the network processes [Bor05]. Then, a centrality value indicates a node's importance with respect to a process with these properties.

While this has been known for several years, only recently have datasets containing real-world network flows become accessible. In this context, the goal of this dissertation is to provide a better understanding of the *actual* behavior of real-world network processes, with a particular focus on centrality measures: If real-world network processes turn out to show different properties than those assumed by classic centrality measures, these measures might considerably under- or overestimate the importance of nodes for the actual network flow. To the best of our knowledge, there are only very few works addressing this topic. The contributions of this thesis are therefore as follows: (i) We investigate in which aspects real-world network flows meet the assumptions contained about them in centrality measures. (ii) Since we find that the real-world flows show considerably different properties than assumed, we test to which extent the found properties can be explained by models, i.e., models based on shortest paths or random walks. (iii) We study whether the deviations from the assumed behavior have an impact on the results of centrality measures. To this end, we introduce *flow-based* variants of centrality measures which are either based on the assumed behavior or on the actual behavior of the real-world network flow. This enables systematic evaluation of the impact of each assumption on the resulting rankings of centrality measures. While—on a large scale—we observe a surprisingly large robustness of the measures against deviations in their assumptions, there are nodes whose importance is rated very differently when the real-world network flow is taken into account. (iv) As a technical contribution, we provide a method for an efficient handling of large sets of flow trajectories by summarizing them into groups of similar trajectories. (v) We furthermore present the results of an interdisciplinary research project in which the trajectories of humans in a network were analyzed in detail. In general, we are convinced that a process-driven perspective on network analysis in which the network process is considered in addition to the network representation, can help to better understand the behavior of complex systems.

Acknowledgments

This work would not have been possible without the support of many people who I appreciate a lot. First, I would like to express my gratitude to my supervisor [Katharina Zweig \(Nina\)](#) for all she has taught me as well as for her support in the last years. Although we have been working together for a while now, each discussion with her brought me new insights and perspectives.

I would also like to thank my current and former group colleagues. I really enjoyed our work-related and non-work-related discussions during our lunch and coffee breaks. I will keep the coffee rounds on the 6th floor in good memory! I appreciated a lot having you around: [Wolfgang Schlauch](#), [Mohammed Abufouda](#), [Sude Tavassoli](#), [Marsha Kleinbauer](#), [Sujay Muramalla](#), [Maryam Amir Haeri](#), [Tobias Krafft](#), and [Marc Hauer](#). And also, of course, our neighbors from the other end of the corridor: [Raphael Reitzig](#), [Sebastian Wild](#), and [Thomas Daun](#). I would also like to thank the students with whom I worked in recent years, particularly [Paul Fröhling](#). Furthermore, I would like to thank [Ingrid Romani](#), not least for supporting all of us with administrative issues.

Furthermore, I would like to express my thanks to [Olaf Peters](#), who approached me with an interesting idea for an interdisciplinary research project. Although most of our results are not included in this thesis, I really enjoyed our discussions and the joint development of new ideas.

Last, but not least, I would like to thank my partner [Michael](#) for his patience and his support in the last years, and particularly in the last months. Writing a PhD thesis while having a small child is not an easy task—writing a PhD thesis during Corona times when childcare is closed for several months would not have been possible without his absolute support!

Contents

List of Figures	ix
List of Tables	xi
1 Introduction	1
1.1 Motivation	2
1.2 Main contributions	4
1.3 Thesis outline	6
2 Background and related work	9
2.1 Complex networks	9
2.2 Network processes	16
2.3 Empirical analysis of network flows	21
3 Properties of network flows in complex networks	25
3.1 Motivation	25
3.2 Datasets used	27
3.3 From where to where does it flow?	31
3.4 How does it flow?	38
3.5 Can properties of trajectories be explained by a random walk model?	45
3.6 Summary	53
4 Flow-based centrality measures	55
4.1 Motivation	55
4.2 Flow-based centrality measures	58
4.3 Comparison methods for rankings	70
4.4 Datasets used	75
4.5 How robust are standard centrality measures?	78
4.6 Which nodes are impacted?	84
4.7 A note on the results' sensitivity to changes of the trajectory set	103
4.8 Which assumptions matter, which don't?	106
4.9 Summary and limitations	108
5 Summarizing trajectories of real-world network flows	111
5.1 Motivation	111
5.2 Similarity and distance measures for walks in graphs	113
5.3 Clustering of trajectories	126
5.4 Limitations and summary	139
6 Case Study: Analyzing game trajectories	143
6.1 Motivation and background	144
6.2 Error categorization system	146
6.3 Experimental setup	154

6.4	Analysis of experiment results	156
6.5	Summary	167
7	Towards a process-driven network analysis	169
7.1	The relevance of network processes for network analysis	169
7.2	Thinking about network analysis from a process perspective	171
8	Summary	173
8.1	Future work	175
A	Supplementary material	177
A.1	Supplementary material for Chapter 3	177
A.2	Supplementary material for Chapter 4	179
A.3	Supplementary material for Chapter 5	184
	Bibliography	187
	List of abbreviations and notations	201
	Publications	207

List of Figures

1.1	Phases of a network analytic project	3
1.2	An overview of the terminology used.	4
1.3	Why taking into account network processes can be helpful	4
3.1	Overview of the datasets used in this chapter.	28
3.2	Network coverage by the real-world network flow trajectories.	32
3.3	Node usage distribution of the real-world network flows	33
3.4	(Absolute) node usage by real-world network flows, compared to values of common centrality measures.	34
3.5	Source-target-frequency of real-world network flows.	35
3.6	The Gini index and the corresponding Lorenz curves for measuring the equality of the distribution of the total amount of flow among all (used) node pairs.	37
3.7	Source-target frequency by distance of the nodes	39
3.8	Path lengths	41
3.9	Analysis regarding equal use of alternative routes between node pairs.	43
3.10	Coverage of the networks by real trajectories, shortest paths, and random walks	49
3.11	Node usage by real trajectories and by random walks.	51
3.12	Cumulative node usage distribution of real trajectories and random walks	52
3.13	Cumulative distribution of source-target-frequency of random walks and real trajectories	52
4.1	Example network flow	57
4.2	Flow-based betweenness measures on an example graph	60
4.3	Flow-based closeness centrality measures on an example graph	65
4.4	Explanation for the introduced measure weighted overlap τ_w	74
4.5	Overview of the datasets used.	76
4.6	Example for two different representations of the same transportation system.	78
4.7	Values and ranks of flow-based betweenness variants compared to standard betweenness centrality	80
4.8	Values and ranks of flow-based closeness variants compared to standard closeness centrality.	81
4.9	Betweenness values on a map for DB1B	85
4.10	Betweenness values on a map for London Transport	86
4.11	Top 10 nodes of flow-based betweenness variants.	87
4.12	Extracts of the air transportation network of the DB1B dataset and the lines network of London Transport dataset.	88
4.13	Lines network representation of London Transport	90
4.14	Top ten nodes of flow-based closeness variants	92
4.15	Values of flow-based closeness variants on map (DB1B), US	93
4.16	Values of flow-based closeness variants on map (DB1B), Alaska and Hawaii	94
4.17	Values of flow-based closeness variants on map (London Transport)	94
4.18	Min-max-plot of rankings for flow-based betweenness variants.	97

4.19	Comparison nodes rankings with external node attributes (DB1B)	99
4.20	London Transport: Span of ranking positions versus external node attributes	101
4.21	Comparison nodes rankings with external node attributes (Wordmorph)	104
4.22	Effect of preprocessing steps on the results of flow-based betweenness measures	105
5.1	State space of a board game and human navigation in it	112
5.2	Overview of how a walk in a graph can be modeled	115
5.3	Examples of walks where the set-based measures are not satisfactory.	116
5.4	Which choices of mapping relations and of which aggregations in the cost function lead to which distance measures	120
5.5	Modeling of walks as sets and sequences of points in a metric space and possible resulting distance measures.	122
5.6	Example where the triangle inequality for δ_{adF} and δ_{adF}^N fails.	125
5.7	A Rush Hour game configuration and its resulting state space. The goal is to arrange the cars in such a way that the red car can leave through the exit on the right side.	128
5.8	Procedure for evaluating measures introduced in Section 5.2	130
5.9	Illustrating the introduced measures weighted and unweighted average purity of a clustering.	132
5.10	Illustration for computing the relative AUC	132
5.11	Unweighted and weighted average purity of the clustering results for exemplary configurations	134
5.12	Relative AUC of weighted and unweighted average purity for all walks of all configurations	137
5.13	Distribution of walk length in 2 clusters	138
5.14	Examples where clustering separates shorter and longer walks	139
5.15	Relative AUC of weighted and unweighted average purity, including only sufficiently long walks	140
6.1	Rush Hour board configurations for illustrating the concept of equivalent configurations	147
6.2	Validation of error category system: number of error moves in benchmark dataset for each game configuration	152
6.3	Validation of the error category system: How many of the error moves fall into how many categories?	153
6.4	Validation of error category system: Percentage of uncategorized error moves of the benchmark dataset, dependent on the difficulty of the game	153
6.5	Design and procedure of the experiment	155
6.6	Experimental data: Frequency of occurrence of each error category	158
6.7	Experimental data: Location of errors in the solution	160
6.8	Experimental data: Proportion of occurrence of each error category	162
6.9	Decision trees predicting the presence of an error in the second solution attempt	167
A.1	Ten nodes used most frequently by real trajectories and random walks.	178
A.2	Overlap of each flow-based betweenness measure to standard betweenness centrality	179
A.3	Ranking positions of the nodes by flow-based betweenness measures compared to standard betweenness centrality	180
A.4	Ranking positions of the nodes for flow-based closeness measures compared to standard closeness centrality	181
A.5	Ranking behaviour of top 10 nodes of flow-based in-close	182
A.6	Min-max-plot of rankings for flow-based closeness variants.	183

List of Tables

2.1	Centrality measures and their assumptions about the network flow [Bor05]	18
2.2	Catalog of network process properties that determine the way an entity uses the network structure. The first two properties, transmission mechanism and trajectory type, were proposed by Borgatti [Bor05].	21
3.1	Basic properties of the datasets used	30
3.2	Spearman's correlation coefficients of classic network measures to actual node usage by real-world trajectories	34
3.3	Five most frequently used source-target pairs for the datasets DB1B, London Transport, and Wikispeedia	38
3.4	Trajectory types of the real trajectories	40
3.5	Number of node pairs used for the analysis of route alternatives	42
4.1	Flow-based betweenness centralities	60
4.2	Flow-based closeness centralities	66
4.3	Example of different ranking methods	71
4.4	Basic properties of the used datasets	75
4.5	Spearman rank correlation and weighted overlap between each flow-based betweenness measure and the standard betweenness measure	82
4.6	Spearman rank correlation and weighted overlap between each flow-based closeness measure and the standard closeness centrality	83
4.7	Span of ranking positions of flow-based betweenness and closeness measures	84
4.8	Flow-based betweenness: Most stable and unstable nodes for dataset DB1B	98
4.9	Flow-based betweenness: Most stable and unstable nodes for London Transport dataset (lines graph)	100
4.10	Flow-based betweenness: Most stable and unstable nodes for London Transport dataset (transitive graph)	100
4.11	Flow-based betweenness: Most stable and unstable nodes for Wikispeedia dataset	102
4.12	Flow-based betweenness: Most stable and unstable nodes for Wordmorph dataset	103
4.13	Effect of preprocessing decisions on the results of flow-based betweenness centralities	105
4.14	Violation of assumptions in datasets.	107
4.15	Spearman rank correlation coefficient between values of the flow-based betweenness measures, and weighted overlap τ_w between the rankings of the measures	107
5.1	Similarity and Distance Measures	118
5.2	Overview of the dataset used	129
5.3	Weighted average purities of the clusterings for the introduced measures for a fixed number of clusters and selected game configurations	136
6.1	Examples for conceptual error categories	148
6.2	Validation of error category system: Are there unnecessary categories?	154
6.3	Tasks used in the experiment.	156
6.4	Experimental data: Categorization of error moves according to error category system	157

6.5	Experimental data: Number of solutions containing each error type at least once. . .	158
6.6	Matches of error moves in participants' initial and revised solution and the peer solution	164
7.1	Aspects of classic network science with an example and how they can be transferred to a process-based perspective.	172
A.1	Properties of the similarity and distance measures for walks in graphs, introduced and used in Chapter 5.	184

In the last twenty years, network analysis has emerged as a popular and powerful tool for understanding and analyzing complex systems [WF94, BLM⁺06, BE05, Ves18]. Complex systems consist of many components interacting with each other in various ways. While there are several definitions for complex systems, one of their fundamental properties is that they show *emergent behavior*: a system's behavior which cannot be explained by the properties of the single components, but is a consequence of their interactions. Vicsek phrases this property of a complex system by stating that “the laws that describe its behaviour are qualitatively different from those that govern its individual units” [Vic02].

A complex system can be represented as a network [New10]: The system's components are represented as *nodes*, their interactions are represented as *edges*. This representation as a graph structure has great potential for enabling a better understanding of complex systems and has been proven to be useful in various disciplines, such as—to name but a few—in sociology for understanding social systems [BMBL09], in biology for understanding the functionality of a cell by studying the interactions of molecules [BO04], in medicine for identifying disease genes [BGL10], or in economy for understanding the interdependencies of financial institutions [SFS⁺09].

In recent years, a wealth of methods, measures, algorithms, and software tools have been developed that make it very easy to create a network representation and apply network analytic methods on it—for example to identify the system's most important components through centrality measures [KLP⁺05], to find cohesive groups of system components through community detection algorithms [GN02], or to predict the future evolution of a system through link prediction methods [LNK07]. However, it is sometimes overlooked that a network representation and all network analytic methods entail specific assumptions. Representing a complex system as a network allows analyzing *indirect effects* in the system [BRMW13, Zwe16]: An interaction between the entities A and B is modeled as an edge between node “A” and node “B” in the network, and an interaction between the entities B and C yields an edge between node “B” and node “C” in the network. By creating a network representation from these interactions and connecting the nodes “A” and “B” as well as the nodes “B” and “C”, it is assumed that entities A and C have some *effect* on each other via B—or share some *commonality* via the link over B. A network representation and network analytic methods enable the analysis of these *indirect effects*. If indirect effects are not present or not of interest, other types of representation or other analysis methods are more convenient and more efficient.

How can these indirect effects be explained? One possible explanation for indirect effects is that “something” is flowing through the network from node to node by using the existing edges. For example, in social systems, pieces of information or gossip stories can be told from person to person, whereby a person can have an indirect effect on another person without any direct connection; infections or other diseases can be spread from person to person, which also causes indirect effects; entities such as passengers in a transportation infrastructure flowing from node to node are another example of such a network flow yielding indirect effects.

While the network representation itself requires the presence of indirect effects, several network methods implicitly assume the presence of a network process, particularly centrality measures. The most well-known centrality measures were introduced by Freeman with the idea in mind that they quantify a node's importance with respect to a network process: "Thus, the use of these three measures is appropriate only in networks where betweenness may be viewed as important in its potential for impact on the process being examined" [Fre77]. Borgatti points out that the most commonly used centrality measures do not only assume the *presence* of a network process, but also assume specific properties of those processes [Bor05]. He found that most centrality measures either assume a network process with a transfer mechanism (where indivisible entities *move* from node to node) or a process solely using shortest paths. It is clear that most relevant network processes, such as the spreading of information or infections, do not satisfy these requirements. However, even for network processes for which those assumptions seem reasonable, it is not clear whether the assumptions are met when *real-world data* about them is considered.

Since various datasets from different domains are available that contain information on how a real-world network flow moves through a network, several questions arise: Do the real-world network flows show the same properties as the centrality measures assume about them? If not, can a different flow model reproduce the properties of the real-world network flow? But also: if the real-world network flow does not have the same properties as assumed by the centrality measures, does this have an impact on the results of the centrality measures? Furthermore, when dealing with large datasets of real-world network flows, how can the data be summarized to enable efficient handling of the data?

This thesis will address these questions while the focus of this thesis is on centrality measures, most of which assume network processes using a transfer mechanism and shortest paths. Thus, this thesis is restricted to the analysis of real-world network flows where those assumptions seem reasonable.

1.1 Motivation

A common approach for understanding a complex system by a network analytic approach is the following procedure [Zwe16] (shown in Figure 1.1):

- (i) The behavior of the complex system is observed in order to obtain data about the entities' interactions.
- (ii) Observations of interactions of the system are used to build a network representation.
- (iii) Standard network analytic methods are applied to the network representation.
- (iv) The results of these methods on the graph representation are then translated back into the original context in order to draw conclusions about the complex system of interest.

Each step appears to be natural, but each has its caveats and uncertainties:

- (i) Observing a system to collect raw data about the system's interactions involves decisions on the part of the researcher, such as the choice of which type of interactions to be observed, and is prone to errors in measurement.
- (ii) Transforming observations of interactions in a complex system into a network representation is rarely unique: for the same set of observed interactions, there are often several plausible network representations. At the same time, it has been shown that seemingly trivial decisions in creating the network representation can have a considerable effect on the resulting network structure and thus also on any network analytic result [But09, DCMHW10].
- (iii) For the same question (such as "Which node is the most important one?" or "Which nodes belong to the same community?"), there is often more than one possible applicable network analytic method: There are dozens of different centrality measures [KLP⁺05], and dozens

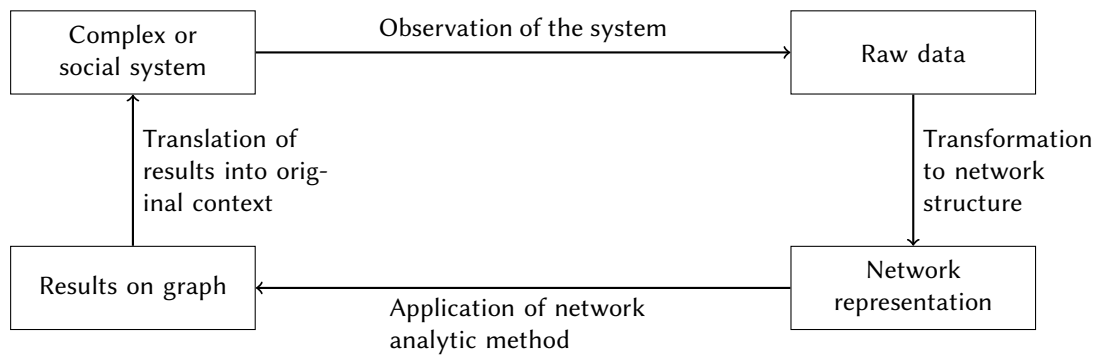


Figure 1.1 Phases of a network analytic project (Figure adapted from Zweig [Zwe16])

of different community detection algorithms [For10]. The choice of the “right” method is not trivial, as even the choice of the normalization method can have an impact on the result [TZ16].

- (iv) When translating the results on the graph representation back into the original domain in order to draw conclusions about the system, all choices need to be considered when interpreting the results since each choice is associated with certain assumptions.

We argue that network analytic approaches where the research question is tied to a network process can benefit from a process-driven perspective in which the focus is on the process of interest. We distinguish the terms *network process* and *network flow*: We understand a network process as a model of how something is flowing through the network whereas the network flow itself consists of observable trajectories of something using the network (see Figure 1.2). Taking into account the relevant network process and network flow is beneficial mainly for the following two reasons:

- (I) Taking into account the *process* of interest during all stages of the network analytic procedure can help to reduce the number of relevant choices in each step (see also Figure 1.3). It has been shown that the process of interest, the network representations and the network measures cannot be chosen independently [DLZ12]: The network process of interest restricts the set of possible network representations, since the network representation needs to be chosen such that the edges represent the relationship *relevant for the process of interest*. When the focus is on a process such as the spreading of a disease, the corresponding network representation needs to contain edges representing relations that are essential for passing on this disease. Network representations where the edges represent a different type of relation, will yield results that are difficult to interpret. Thus, the process of interest constrains the set of possible network representations for a meaningful network analysis. At the same time, the process of interest can restrict the set of applicable network measures: Borgatti argued that in the case of centrality measures, each centrality measure implicitly contains a process *model* with certain properties [Bor05]. A centrality measure can then only quantify a node’s importance with respect to network flows with these properties. Using a centrality measure for which the process model assumes the usage of shortest paths, for measuring a node’s importance for spreading a gossip story, will yield results that are hard to interpret. Therefore, the network process of interest restricts both, the set of possible network representations and the set of applicable network measures [DLZ12]. Thus, the process of interest constrains the set of applicable network measures. At the same time, the chosen network representation affects the results of any applied network measure [But09, DCMHW10]. The interdependencies of the network process, the network representation, and the network measure are depicted in Figure 1.3.
- (II) Representing a system as a network is a strong simplification of a system: A complex system consisting of various components with possibly different individual properties and behaviors,

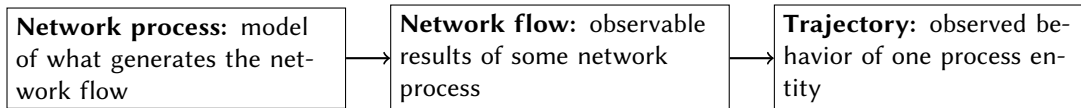


Figure 1.2 An overview of the terminology used.

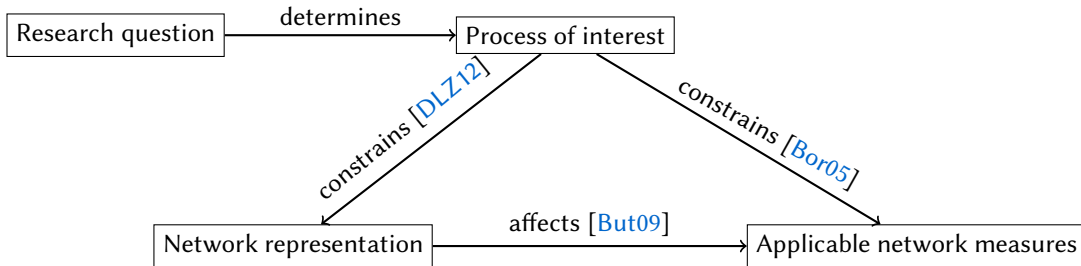


Figure 1.3 Why taking into account network processes can be helpful: If the process of interest, determined by the research question, is considered at the beginning of any network analytic project, it can help to reduce the number of relevant choices in each step of the network analytic project.

interacting in various ways with each other, is transformed into a very restrictive mathematical structure, consisting only of nodes and edges. As with any modeling approach, a lot of information is lost by this transformation. But if such a representation is chosen appropriately, it is able to capture the essential properties of a system—while being simpler and easier to handle. In other words, when chosen appropriately, such a representation strips off all unnecessary information and only focuses on the relevant information required to answer the research question of interest. It is, however, not clear whether a static network representation is sufficient in all cases to capture the essential properties of a system. We argue that, particularly when the system is the infrastructure for a process disseminating through the system, it might be valuable to consider the process dynamics in addition to the static network representation. Consider the air transportation system [GMTA05] where each city area is represented by a node, and existing airline connections are represented by an edge. A relevant flow in this network is the flow of passengers traveling from a start city to their destination. By solely considering the network representation, the actual passenger flow is neglected: The edge between two nodes representing small rural cities has the same quality as an edge between nodes representing large metropolises. This can lead to anomalies with respect to the nodes' centrality scores [GMTA05, DLZ12]. When taking into account the actual passenger flow, it becomes obvious that those example edges are of different qualities. It is therefore important to not only consider the static network representation, but also—if available—the actual process dynamics on the network structure itself.

1.2 Main contributions

Based on the motivational aspects stated above, this thesis addresses the analysis of network flows and their connection to network measures, where the focus is on centrality measures. In this context, the following contributions are made.

Properties of real-world network flows Particularly centrality measures assume the presence of a network flow with specific properties and measure the nodes' importance with respect to network flows with these properties. It is, however, not clear whether these properties are matched by existing processes. Therefore, we considered datasets containing trajectories of real-world network flows, two datasets containing passenger flows in transportation systems, and three datasets containing human navigation trajectories in game-like contexts. For these

real-world network flows, we can answer the following question:

Do real-world network flows show the same properties as assumed by certain network measures?

Therefore, the properties of real-world network flows are compared to their assumed properties. Besides the usage of shortest paths, centrality measures expect the network flow to be uniform in the sense that there is an equal amount of flow between each node pair. We show that this is not given in all considered datasets—on the contrary, for real-world network flows, there are a few node pairs between which there is a large proportion of the total flow, while there is (almost) no flow between many other nodes. A further assumption contained in certain centrality measures concerns the behavior of the flow when there are several equivalent possibilities to flow from node A to node B. In this case, measures normally assume that the flow is distributed among the alternative routes. We investigate whether this behavior is also shown by real-world network flows.

Alternative process models Since we show that real-world network flows show different properties than assumed by standard centrality measures, but datasets containing real-world network flows are rarely available, there is a need for an alternative process model. If such a model was available, it could be used to make more accurate predictions for flow-related problems than the standard process model. The main question is thus:

Can the real-world network flow be simulated to a satisfying extent by a model?

In order to answer this question, we investigate whether the properties of the real-world network flow can be reproduced by a model based on shortest paths or by a model based on random walks. For this reason, several model variants are implemented and the properties of the trajectories generated by these models are compared to the properties of real-world network flows.

Impact on centrality measures Since real-world network flows show different properties than the process model implicitly contained in standard centrality measures, a naturally arising question is whether this actually matters; i.e.:

Which impact does it have on the results of centrality measures that their assumptions about the network process are not satisfied?

In order to answer this question, we introduce "flow-based" variants of the classic centrality measures, namely closeness and betweenness centrality: these variants either use the property of the standard process model or the property of the actual real-world network flow contained in the flow datasets. For each assumed property (for example usage of shortest paths), the flow-based variants allow "switching on or off" the assumption or the actual flow behavior. This approach enables systematic evaluation of the importance of each contained assumption. We find that at a large scale, the considered centrality measures are surprisingly stable against deviations in their process model. At the same time, we observe large ranking variations of single nodes among the centrality variants. Those large ranking variations even affect high-ranked nodes.

Summarizing a large set of flow trajectories Since we find that simple models are not able to explain the properties of real-world network flows, it is necessary to consider the single trajectories of real-world network flows in order to understand the underlying system. However, when dealing with a large number of trajectories in a graph, limited computing resources might constrain a detailed analysis. For this reason, we aim for a method for grouping and summarizing trajectories. Then, an analysis can be carried out on a smaller set of representative trajectories instead of on the full set of data:

Given the data of a network flow with transfer mechanism, how can the trajectories be summarized enabling a more efficient analysis?

For this reason, in Chapter 5, several similarity and distance measures for walks on graphs are introduced which are then used to group trajectories using a standard clustering procedure. We provide a taxonomy of modeling variants for walks, review existing similarity measures from different domains and adapt them to the case of walks in graphs¹. These similarity and distance measures are then evaluated by their ability to find meaningful groupings of trajectories, by using a dataset containing trajectories with ground truth.

Domain-specific analysis of flow trajectories While the challenges and corresponding approaches above described are rather concerned with network flows of different domains, we describe the results from an interdisciplinary collaboration project in which domain-specific trajectories in a network were analyzed in detail:

How can trajectories in networks of a specific domain be analyzed in detail in order to get insights for the domain?

In this case, we describe the results of a joint research project with researchers from the Department of Psychology at the Technical University (TU) of Dresden. For their research question in the context of learning and human problem solving, an experiment was designed and conducted in which participants were asked to solve a sliding-block puzzle. A participant's solution attempt could then be modeled as a walk through the game's state space which opens up new methods of analysis. In this thesis, we propose a novel approach for analyzing such human problem solving attempts by introducing an error category system which allows new insights in the cognitive process of human problem solving.

1.3 Thesis outline

This thesis is structured as follows:

- Chapter 2 sets the context of this thesis by presenting publications that are relevant for this work, and also introduces necessary definitions and notations.
- In Chapter 3, the properties of real-world network flows are tested using network flow datasets. Particularly those properties assumed by classic centrality measures are tested. Chapter 3 also includes an analysis of whether a random-walk-based model can explain the properties of real-world network flows. The results presented in this chapter are mainly based on publications [5] and [8].

¹In a previous work [Boc15], a preliminary set of similarity and distance measures was proposed which is extended in this thesis.

- Based on the findings of Chapter 3, in Chapter 4, we investigate which impact it has on the results of classic centrality measures that its basic assumptions are not fulfilled. The results of this chapter are mainly based on publications [4] and [6].
- Chapter 5 presents an approach for summarizing and grouping a set of trajectories into representative groups. The results of this chapter are based on publications [3] and [2].
- Chapter 6 presents the results of the interdisciplinary collaboration project in which a detailed analysis of trajectories in a graph was used to get a better understanding of the cognitive process of human problem-solving. The results of this chapter are based on publication [1] and represent joint work done with Olaf Peters and Susanne Narciss from the Department of Psychology at TU Dresden.
- In Chapter 7, an extended outlook is given in which the importance of considering network processes in addition to the static network representation is emphasized.
- Finally, Chapter 8 gives a summary of the results presented in this thesis and sketches ideas for future work in this area of research.

Background and related work

Chapter outline

In this chapter, the context of this thesis is set. For this reason, we will give a short introduction to network analysis, provide the definitions needed in this thesis, and review related work that is relevant for this thesis.

2.1 Complex networks

Complex network analysis is a methodology for understanding and analyzing the behavior of complex systems [New02]. A complex system consists of independent entities interacting with each other such that the system exhibits a so-called emergent behavior, a behavior that cannot be explained by the behavior of the single entities, but only by their interactions. Network analysis enables the analysis of these interactions by representing the system as a network in which the system's entities are represented as nodes and their interactions are represented as edges. Mathematically, networks are modeled as *graphs*. We distinguish the terms *graph* and *network*: While a graph refers to a mathematical object consisting of nodes and edges, a network consists of nodes and edges, but these *represent* the system's entities and their interactions. Therefore, we use the term network to refer to a graph structure with a mapping between the nodes and the system's entities.

As all network analytic methods and tools are defined on the graph structure, we will introduce the necessary definitions and notations in the following.

2.1.1 Graph definitions

Definition 2.1: Graph

A graph $G = (V, E, \omega)$ consists of a set of nodes $V = \{v_1, v_2, \dots, v_n\}$, a set of edges $E \subseteq V \times V$, and a weight function $\omega : E \rightarrow \mathbb{R}$.

If the edge set E consists of unordered pairs $\{v, w\}$ with $v, w \in V$, the graph is said to be *undirected*. If the edge set E consists of ordered tuples (v, w) , the graph is called *directed*. The weight function ω assigns weights to the edges. If the weight function ω is the trivial function $\omega(e) = 1$ for all $e \in E$, the graph is called *unweighted*, and the notation can be simplified to $G = (V, E)$. Otherwise, the graph is called *weighted*. In most cases, the edge set E is a simple set allowing at most one edge from node v to node w and does not contain edges such as (v, v) (so-called self-loops), yielding a *simple graph*. In some cases, multi-edges or self-loops are needed, then E is a multi-set allowing multiple edges from node v to node w and the graph is called a *multiple graph*. In this work, we

consider simple, directed, weighted graphs.

If the graph is directed, v is called the source node of the edge $e = (v, w)$, and w is called the target node of the edge. An edge $e = (v, w) \in E$ is called *incident* to the nodes v and w . If the edge set contains the edge $e = (v, w) \in E$, then, the nodes v and w are called *adjacent* to each other or are said to be *neighbors* of each other. For a directed graph, we distinguish the *out-neighbors* (or *successors*)

$$N^{\rightarrow}(v) = \{w \in V \mid (v, w) \in E\} \quad (2.1)$$

of node v and the *in-neighbors* (or *predecessors*)

$$N^{\leftarrow}(v) = \{w \in V \mid (w, v) \in E\}. \quad (2.2)$$

The number of neighbors of a node v is called the *degree* of a node, denoted by $\text{deg}(v)$. For directed graphs, a distinction is made between a node's in-degree and its out-degree, denoted by $\text{deg}^{\leftarrow}(v)$ and $\text{deg}^{\rightarrow}(v)$, respectively, as the number of its in-neighbors and as the number of its out-neighbors.

Definition 2.2: Subgraph

A subgraph $G' = (V', E')$ of graph $G = (V, E)$ is a graph with $V' \subseteq V$ and $E' \subseteq V' \times V' \subseteq E$. An induced subgraph $G' = (V', E')$ is a subgraph of G where $V' \subseteq V$ and $E' = \{(v, w) \in E \mid v, w \in V'\}$, i.e., G' must contain all edges of E for which source and target is in V' .

Definition 2.3: Walks, trails, and paths

A walk is an alternating (finite) sequence of nodes and edges, $P = (v_1, e_1, v_2, \dots, e_{k-1}, v_k)$ with $v_i \in V$ and $e_j = (v_j, v_{j+1}) \in E$ for all $i \in \{1, \dots, k\}$ and $j \in \{1, \dots, k-1\}$, respectively. If the edges of P are pairwise distinct, P is called a trail. If nodes and edges of P are pairwise distinct, P is called a path.

In this work, only simple graphs are considered, thus, P is uniquely determined by its node sequence and the notation can be simplified to $P = (v_1, v_2, \dots, v_k)$. The *length* of a walk P is denoted by $|P|$ and is defined as

$$|P| = \sum_{i=1}^{k-1} \omega(e_i). \quad (2.3)$$

The start node (or source node) of the walk P is denoted as $s(P) = v_1$, the end node (or target node) as $t(P) = v_k$. The i -th node in the node sequence of P is denoted as $P(i) = v_i$ for $1 \leq i \leq k$. If a node v is contained in a walk P , we write $v \in P$. Similarly, if an edge e is contained in a walk P , we write $e \in P$. The set of nodes contained in a walk P is denoted by $V(P)$, the set of edges contained in a walk is denoted by $E(P)$. Note that $|V(P)| \leq k$. We furthermore define the index set of a walk by $I(P) = \{1, 2, \dots, k\}$. Let $P_{i,j}$ for any $1 \leq i \leq j \leq k$ denote the sub-path of P with $P_{i,j} = (v_i, v_{i+1}, \dots, v_j)$.

Definition 2.4: Graph distance

For two nodes $v, w \in V$, the length of the shortest path from v to w is called the *distance* from v to w and is denoted by $d(v, w)$. If there exists no path in the graph from v to w , it is set $d(v, w) := \infty$.

The longest shortest path between any two nodes in the graph is called the *diameter* of a graph, $\text{diam}(G) = \max_{v, w \in V} d(v, w)$. For a node v and a walk P in G , let $d(v, P)$ denote the length of the shortest path from v to any node in P , i.e., $d(v, P) = \min_{w \in V(P)} d(v, w)$.

Definition 2.5: Connectedness

An undirected graph is called connected if there exists a path between any two nodes in the graph. A directed graph is called strongly connected if for any two nodes $v, w \in V$, there exists a path from v to w . A directed graph is called weakly connected if replacing all directed edges in G by undirected edges yield a connected (undirected) graph. A connected component of a graph G is a maximal connected subgraph of G .

Definition 2.6: k -core

A maximal connected subgraph in G in which all nodes have a degree of at least k , is called a k -core. A node v has a *coreness* k if it belongs to a k -core, but not to a $(k + 1)$ -core.

2.1.2 A brief history of complex network analysis

In order to understand the context in which network analysis and its methods evolved, it is useful to briefly review the historical development of this discipline. The theory of graphs dates back to the 18th century when Leonhard Euler published a mathematical problem that became famous under the name *Seven Bridges of Königsberg* [Eu41]: Through the city of Königsberg, a river flows in which there are two islands. The city areas on both sides of the river and the islands are connected by several bridges. The question was: Is it possible to walk through the city using each bridge exactly once? By transforming the city into a graph structure in which a node is a city area and an edge is a bridge connecting those areas, it is easier to prove that such a walk cannot exist. Although Euler did not directly use this representation, Euler's contribution is the introduction of a new level of abstraction that made it easier to solve the problem, and which is why this work is seen as the beginning of graph theory. From then on, the discipline of graph theory evolved, where the structural properties of graphs were proven and methods for solving problems on graphs were developed. Examples include questions such as: How many colors are needed when each node of a graph shall be colored such that any two adjacent nodes are not colored with the same color (graph coloring problems)? How many graphs with certain properties exist (graph enumeration problems)? Methods were also developed for deciding whether a graph contains a certain subgraph (graph isomorphism problems) or for the efficient computation of the shortest paths.

While the field of graph theory is concerned with the structural properties of graphs as mathematical objects, the field of network analysis uses graph structures as a representation format in order to analyze the underlying system. One of the first historical examples where a network representation was used to infer knowledge about the underlying system, was given by Sylvester [Sy178] who used a graph for representing the structure of chemical molecules in the 19th century. Sociologists constructed so-called *sociograms* (network representations in our terminology) for understanding social systems: In these sociograms, persons are represented as nodes and their relationships are represented as edges. In the 1930s, Jacob L. Moreno and his colleagues were among the first ones using these forms of representation [MJ38, Mor77] to understand phenomena in social groups.

Over the years, a large toolbox of methods for analyzing the structure of a network and gaining insights about the underlying system has been developed, motivated by questions in different contexts, such as:

- **How can the structure of the network as a whole be characterized?** To answer this question, a lot of measures exist, quantifying some aspect of the network structure, for example the size of the network (its number of nodes), the density of the network (the number of its edges divided by the number of possible edges), the transitivity and clustering coefficient (both measures for quantifying to which extent the network is locally dense, i.e., whether

there exist dense substructures), as well as measures based on the nodes' distance to each other, for example, the network's diameter or the average path length (the mean distance between all pairs of nodes).

- **Which nodes are the most important ones with respect to their position in the network?** This question can be answered with the help of centrality measures [KLP⁺05]. Section 2.1.3 is dedicated to introducing existing centrality measures that are relevant for this thesis.
- **Which system entities have similar roles in the system?** Consider for example a network representing the social interactions between members of a company or an organization. If the social structure of each department is built similarly, methods of (structural) equivalence aim at identifying positions with similar functions in the system [LW71], for example, the secretary or the department head.
- **Which system entities belong together?** Particularly in social network analysis, identifying groups of persons is of interest. For example, when considering who is a friend of whom, there are groups of highly connected people, with loose connections to other groups of friends. Identifying these groups through the network structure is the goal of clustering methods [GN02, DPV05, For10].
- **Do system entities form recurring patterns?** Particularly in biological networks, for example interaction networks of two biological components such as mRNAs and microRNAs, it has turned out that small subgraphs with a certain structure occur extraordinarily often [Mil02]. This is why it is likely that those structures (so-called motifs) provide a certain functionality in the system. Methods for finding such patterns in the network have been used for identifying potential tumor suppressors for breast cancer [UMZ⁺12].
- **How will the system evolve in the future?** The previously mentioned methods are applied to the network representation of a system at a certain point in time. However, systems often change over time: entities join or leave the system, new interactions are started or existing interactions are ended. Link-prediction methods aim at predicting the future evolution of the system based on its current (or past) structure [LNK07].

Although networks were known in various disciplines (under various names) and methods and measures for analyzing them were developed, until the end of the 20th century, there did not exist a research area dedicated to the analysis of networks per se. Networks were rather used as a form of representation for systems of the respective discipline. This changed at the end of the 20th century when personal computers began to be a more common tool also for research. Until then, manually collected and curated network representations of systems were mainly analyzed manually which made it infeasible to consider networks larger than a few nodes. The availability of personal computers made automated analysis of networks possible which is why more networks and networks of a larger size could be considered.

During this time, two articles were published that gave rise to the discipline of complex network analysis as we know it today. Instead of considering the structure of *one* network in order to understand the underlying system, these works consider the structure of *several* networks of different domains. One paper is by Barabási and Albert [BA99] who considered the network representations of systems from several domains, for example a network representation of the World Wide Web (at that time), of the power grid system in the United States, or the citation network where a node represents a published paper and an edge represents a citation. Their surprising finding was that the structure of the networks showed similar properties. They considered the networks' degree distributions and found that the degree distributions of all considered networks showed so-called scale-free behavior, i.e., the distributions followed a power law. This means that there are a few highly connected nodes while most nodes have a very small degree. This is surprising for two reasons: (i) Since these systems were formed by different mechanisms, different structural properties could have been expected. (ii) Until then, random graphs such as the famous $\mathcal{G}(n, p)$ or $\mathcal{G}(n, m)$ [ER60] were used as a model for unknown network structures. In those random graph models, the degrees follow a Poisson distribution [Bol01]—which is very different to a power-law

distribution.

The second seminal paper in the early days of network analysis is the one by Watts and Strogatz [WS98]: Until then, it was assumed that network representations of complex systems can be approximated by either of two graph models: random graphs, such as $\mathcal{G}(n, p)$, or regular graphs where nodes are thought of being placed on a lattice and each node is connected to its direct lattice neighbors. Watts and Strogatz also used the availability of several datasets containing network representations of systems from different domains and considered the average path length as well as a newly introduced measure, the clustering coefficient of these networks. When comparing these two structural properties of the real-world networks to the properties of the regular graph and the random graphs, they found that neither model showed the same characteristics as the real-world networks: While regular graphs exhibit a high clustering coefficient and a large average path length, random graphs show small clustering coefficients and small average path lengths. Real-world networks, however, often have a high clustering coefficient and small characteristic path lengths at the same time. Motivated by this observation, Watts and Strogatz introduced the famous *Small world model* where a parameter p controls how many random edges are inserted into a regular graph such that with increasing values of the parameter p , the generated graphs approach a graph generated by a pure random procedure. Interestingly, there is a range of the parameter p for which the generated graphs show the same properties as the real-world network: a high clustering coefficient together with a small average path length.

2.1.3 Centrality measures

In this thesis, we are mainly concerned with one particular type of network measures, namely centrality measures. Thus, the following section will briefly introduce the most common centrality measures. One question that occurs frequently in network analysis is concerned with the *position* of the nodes within the network: Which node is the most important node due to its position in the network? This can be relevant in various scenarios: In marketing, it can be of interest to determine persons who have a large impact to promote new products; in an epidemic scenario with a limited number of vaccination doses available, it might be of interest to identify persons whose vaccination will have the largest impact on the spread of the disease; for infrastructure planning, it might be of interest to identify the most essential waypoints of a system.

This has led to the notion of centrality. Since the first mention of structural centrality by Bavelas [Bav48], a large number of centrality measures have been proposed because there are different aspects why a node can be considered as being central in the network. Borgatti et al. named several of these aspects [BEJ13]. A node can—for example—be regarded as important because removing the node from the network would affect the network. A node can also be regarded as central because it is connected to a large number of other nodes. Freeman [Fre78, FRM79] reviewed existing centrality measures and distinguished between three aspects on which a node's centrality can be based: (i) its connectedness, through which a node has a large potential of immediate influence on other nodes; (ii) its role as mediator, through which a node has a large potential of control over the communication of other nodes; and (iii) its closeness to other nodes, through which a node has a large potential of indirect influence on other nodes. Based on these three aspects, Freeman proposed a set of measures that are still among the best-known centrality measures: degree centrality, betweenness centrality, and closeness centrality.

The following paragraphs will introduce the most common centrality measures that aim at capturing the importance of a node with respect to different aspects. Common to all of them is that they are functions assigning a value to each node, based only on the structure of the graph. An overview of existing centrality measures is provided by Koschützki et al. [KLP⁺05], a classification of centrality measures based on their computation is given by Borgatti and Everett [BE06].

Degree centrality The degree centrality [Nie74, Fre77] is designed to measure a node's direct connectedness to other nodes in the network, by simply counting the number of a node's incident edges:

Definition 2.7: Degree centrality

For a node v , the degree centrality $D(v) = deg(v)$ is simply defined as the degree of node v . For comparing the degree centrality values of different graphs, the degree centrality is sometimes normalized by its possible maximum value $|V|-1$, yielding the *normalized degree centrality* (denoted by a superscript N)

$$D^N(v) = \frac{deg(v)}{|V|-1}. \quad (2.4)$$

For directed graphs, we can distinguish between the in-degree centrality and the out-degree centrality, counting the in-neighbors and the out-neighbors of a node respectively.

The degree centrality measures the potential *direct* effect of a node since it only counts the direct connections of a node to other nodes.

Closeness centrality A common motivating example for closeness-like centrality measures is a facility location problem: Consider an environment, such as a city, with geographic distances between different positions. A facility such as a supermarket or a hospital, needs to be placed at some position in the environment. There are different possible optimization criteria for the optimal facility location. One example: the total distance of the facility from all other positions in the environment is minimal which leads to the classic closeness centrality. Based on the ideas of Bavelas [Bav50], Beauchamp [Bea65], and Sabidussi [Sab66], Freeman [Fre78] defined closeness centrality as

$$C_F(v) = \frac{|V|-1}{\sum_{w \in V} d(w, v)} \quad (2.5)$$

A node is thus considered as central, i.e., it is a good position for a facility, if the average distance from all other nodes to it is small. There are a few issues with this original definition which is why we will use a slight adaptation. Since in a directed graph $d(v, w)$ is not necessarily equal to $d(w, v)$, we will define two variants, an in-closeness counting the distances *to* node v , and an out-closeness, counting the distances *from* v . If there are nodes for which there exists no path between them, their distance is usually set to ∞ which is problematic with the formula above. We will thus use the following definition:

Definition 2.8: Closeness centrality

For a node $v \in V$, its in-closeness centrality is defined as

$$C^{\leftarrow}(v) = \sum_{w \neq v \in V} \frac{|V|-1}{d(w, v)} \quad (2.6)$$

and its out-closeness centrality is defined as

$$C^{\rightarrow}(v) = \sum_{w \neq v \in V} \frac{|V|-1}{d(v, w)} \quad (2.7)$$

with $d(v, w)$ the length of the shortest path from v to w , $v, w \in V$, and the convention $\frac{1}{\infty} = 0$.

Betweenness centrality Betweenness centrality was independently introduced by Freeman [Fre77] and by Anthonisse in an unpublished work [Ant71]. It is supposed to measure whether a node is positioned *between* other nodes: Nodes that due to their position are able to control the communication between other nodes, so-called gatekeepers, are thought to be in a powerful position since they have the power to interrupt any flow by not forwarding it. This idea is quantified by the following formula:

Definition 2.9: Betweenness centrality

For a node v , its betweenness centrality is defined as

$$B(v) = \sum_{\substack{s \in V, \\ s \neq v}} \sum_{\substack{t \in V, \\ s \neq t \neq v}} \frac{\sigma_{st}(v)}{\sigma_{st}} \quad (2.8)$$

where σ_{st} denotes the number of shortest paths from s to t , and $\sigma_{st}(v)$ denotes the number of those paths containing the node v .

It is thus supposed to measure the amount of flow that a node is able to control by being able to stop the flow when not forwarding it. A slightly different variant is the betweenness centrality including endpoints:

Definition 2.10: Betweenness centrality including endpoints

The betweenness centrality including endpoints for a node $v \in V$ is defined as

$$B_e(v) = \sum_{s \in V} \sum_{t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \quad (2.9)$$

Feedback centralities Another family of centralities considers a different idea for centrality: A person or another entity can also be regarded as important if they have the potential of influencing other important persons. Therefore, nodes connected to nodes with a high centrality value should also get a high centrality value themselves. This idea leads to several centralities, for example, the Eigenvector centrality.

Definition 2.11: Eigenvector centrality

For a node v , its Eigenvector centrality [Bon72] is defined as

$$E(v) = \frac{1}{\lambda} \sum_{w \in N(v)} E(w) \quad (2.10)$$

where λ is a constant, and $N(v)$ is the set of neighbors of node v .

From the definition, it can be seen that a node's Eigenvector centrality is dependent on the centrality value of its neighbors. The name can be explained as follows. The above equation can be rewritten to the eigenvector equation $Ax = \lambda x$ where A is the adjacency matrix of the graph, a matrix of size $|V| \times |V|$ where $a_{ij} = 1$ if there is an edge from node v_i to node v_j and $a_{ij} = 0$ otherwise. x is an Eigenvector, and λ is the Eigenvalue. Since there can be several Eigenvectors and Eigenvalues that solve the Eigenvector equation, the Eigenvector centrality is regarded as the Eigenvector with the largest Eigenvalue. Thus, the i -th entry in the corresponding Eigenvector is

the Eigenvector centrality of node i .

There are several centralities with a similar motivation as the Eigenvector centrality, for example the Katz centrality [Kat53], or Google's PageRank centrality [PBMW99] which is used by Google's search engine to rank web pages by their relevance. Here, the structure of the web pages is considered as a network where links between pages are represented as directed edges. Also for the PageRank centrality, the centrality of a node is dependent on the centrality of its neighbors, more precisely, on the centrality of the nodes pointing to it. The idea is that a web page should be considered as important if it is referenced by other (and many) important pages.

Definition 2.12: PageRank centrality

For a node $v \in V$, its PageRank centrality is defined as

$$PR(v) = d \sum_{w \in N^{\leftarrow}(v)} \frac{PR(w)}{\deg^{\rightarrow}(w)} + \frac{1-d}{|V|} \quad (2.11)$$

where $\deg^{\rightarrow}(v)$ is the out-degree of node v , $N^{\leftarrow}(v)$ is the set of in-neighbors of v , and $d \in [0, 1]$ (usually called a damping factor).

The intuition of the measure is as follows: Similar to the Eigenvector centrality, the PageRank centrality of a node v is dependent on the centrality of the nodes pointing to v . If, however, node w is pointing to a large number of nodes, w should contribute a smaller value to the centrality of v . Therefore, the contribution of w to its neighbors' centrality is divided equally among all the out-neighbors of w which introduces normalization by the out-degree of w . Additionally, a so-called damping factor d is introduced which can be understood as a probability, and which can be best explained by the model of a random surfer, randomly moving through the network. In each step, with probability d , the random surfer randomly chooses one of the outgoing links and moves to the chosen web page. With probability $1-d$, however, the random surfer is "jumping" to a node, chosen randomly from the set of *all* nodes. The PageRank centrality for a node v can then also be understood as the probability that the random surfer is at node v . Introducing this damping factor overcomes several problems: Otherwise nodes without any out-going edges would possibly get high centrality values since the random surfer cannot leave this node anymore.

2.2 Network processes

The above mentioned methods and measures have been used in various contexts to infer knowledge about the underlying system. However, it needs to be emphasized that the network representation itself, all measures and methods are bound to assumptions. Technically, with a network representation at hand, all listed methods and measures can be applied to it. But when the results of the applied methods are interpreted with respect to the underlying system, the methods' assumptions need to be regarded.

Borgatti discussed these assumptions for the most common centrality measures [Bor05]. He points out that each common centrality measure can be understood as a measure quantifying a node's relevance for something flowing through the network. This aspect was already pointed out by Freeman when he introduced betweenness-based centrality measures [Fre77]:

Thus, the use of these three measures is appropriate only in networks where betweenness may be viewed as important in its potential for impact on the process being examined. Their use seems natural in the study of communication networks where the potential for control of communication by individual points may be substantively rel-

evant.

In this connection, we will use the following terminology: We understand a network process as a model of how something is flowing through the network whereas the network flow itself consists of observable trajectories of something using the network. We will use the term *trajectory* if the empirically observed route of one process entity is referred to, while we will use the term walk (or path or trail) if only the *structure* of the trajectory in the graph is relevant or if it is referred to as a *possible* walk in the graph that is not necessarily taken by a process entity.

Each centrality measure does not only assume the *presence* of a network flow, but also certain properties of the network flow. Borgatti [Bor05] identified two dimensions along which such a process model can differ: the **type of node-to-node transmission** and the **type of trajectories** used. For the latter, he distinguishes between *shortest paths*, *paths*, *trails*, and *walks*: For walks, there is no restriction of multiple node or edge visits; a walk is called a trail, if nodes are contained at most once (while edges might be visited multiple times). A trail is called a path, if no node or edge is contained more than once (see also Definition 2.3). For the dimension of node-to-node transmission, Borgatti differentiates between a *transfer mechanism* and a *duplication mechanism*. For processes with a transfer mechanism, an indivisible process entity literally *moves* from node to node. Therefore, the process entity can only be at one node at one point in time. For processes with a duplication mechanism, the process entities, such as pieces of information or infections, disseminate to the next node while still staying at the current node. The duplication mechanism can be further distinguished into two types: *parallel* and *serial duplication*. With a parallel duplication, the process entity is duplicated onto all neighbors of a node at once, while with a serial duplication, the process entity is duplicated onto the node's neighbors one by one.

2.2.1 Which process properties are assumed by centrality measures?

All introduced centrality measures can be interpreted as measuring a node's importance with respect to a network process. For this interpretation, the different measures additionally contain assumptions about the properties of the flow process. In this thesis, we focus on network processes and their properties with respect to centrality measures. Therefore, we will briefly review the single assumptions of each measure as stated by Borgatti [Bor05], in the following section. Table 2.1 summarizes the identified assumptions.

Degree centrality Degree centrality only considers the number of incident edges of a node; thus, it only measures the *direct* effect of or on a node. Indirect effects such as A affecting C via B are not caught by this measure. If the measure is interpreted with respect to a network process, there are two possibilities [Bor05]: It either assumes a process that can only move one step, or it measures the effect of a flow process within one time step. Borgatti names situated knowledge construction as an example of a process without indirect links, where two nodes share something that is unique to them and cannot spread further. The second possibility listed by Borgatti is to measure the effect of a flow process within one time step. Consider a network where each node can either be infected or not and the infection can spread from node to node. If a certain proportion of nodes is infected at one time point, the degree centrality is directly proportional with a node's risk of getting infected in the next time step.

Borgatti argues that degree centrality is appropriate for processes with a parallel duplication mechanism where the initial distribution over the network is random, or for transfer processes where a process entity is performing an infinitely long random walk through the network [Bor05]. In the first case, degree centrality is an indicator of the risk of being infected in the following step; in the second case, degree centrality is proportional to the expected number of visits by the random walker. When considering for which processes with which types of trajectories degree centrality is appropriate, it is noticeable that for this centrality, all types of trajectories *except* shortest paths are

Table 2.1 Centrality measures and their assumptions about the network flow: Table from Borgatti [Bor05] (reprinted from [Bor05], Copyright (2004), with permission from Elsevier): Which common centrality measures assume which type of network process?

		Transmission mechanism		
		Parallel duplication	Serial duplication	Transfer
Trajectory type	Shortest paths		Closeness	Closeness, Betweenness
	Paths	Closeness, Degree		
	Trails	Closeness, Degree		
	Walks	Closeness, Degree, Eigenvector		

allowed. For processes using solely shortest paths, degree centrality cannot give any interpretable number for the importance of a node.

Closeness centrality By calculating the total distances of all nodes to node v (or its inverse), closeness centrality can be understood as a measure for expected arrival time. If a process entity starts in any of the nodes and moves to v on the shortest path, the closeness score of v indicates the expected arrival time of the process entity at v . Nodes for which all other nodes are quite close, will have a higher closeness score, which is why a process entity will arrive at v earlier—in expectation. In this measure, it is assumed that the process entities only use shortest paths. This implies two further aspects: first, each process entity also has a target to reach; second, it knows how to get there as fast as possible. Borgatti argues that closeness centrality is appropriate for measuring a node’s importance with respect to two types of processes: transfer processes along shortest paths, and parallel duplication processes [Bor05]. He mentions that there are also assumptions about reachability in the measure. Since the measure in its original definition by Freeman [Fre77] given in Equation 2.5 is not able to handle disconnected graphs, it is implicitly assumed that only connected (and for directed graphs, strongly connected) graphs are considered. For disconnected graphs, the measure will yield *undefined* or *infinity* for every node (depending on how the distance between two nodes that are not reachable from each other is defined). Additionally, for a node v , the (potential) traffic from all other nodes to v is considered. This includes two assumptions: First, it is assumed that there is traffic from each node w to v (or: the probability that there is traffic from w to v is equal for each w). Second, since all distances are weighted equally, the amount of traffic from w to v is considered as equally important for all w . Apart from that, all centrality measures that consider graph distances such as closeness centrality, implicitly assume that graph distances are actually a meaningful concept for the network and network flow at hand. This seems to be an odd requirement at first sight. The reasoning is as follows: In networks representing infrastructures such as road networks, graph distances have a clear semantic and it is essentially different whether two nodes have a distance of 10 km or of 1000 km. In other networks, this is not that clear: Friedkin analyzed communication networks in academic contexts [Fri83], and found a so-called *horizon of observability*, a distance beyond which members of the network are not aware of the work of the other member anymore. Therefore, for a member, it is of the same quality whether other members are, e.g., 10 or 20 hops away if both distances are beyond this horizon of observability—both are out of his sight which is why there will never be any targeted flow between these members. It is clear that an untargeted flow such as the spread of information can exist between those members, and for such a flow, it is in most cases more probable that an information reaches the member who is 10 hops away than the one who is 20 hops away (and probably faster). For targeted flows, however, it might occur—depending on the flow process at hand—that each flow trajectory has a maximal reachability bounded by the horizon of observability. Since closeness centrality assumes a targeted flow, this aspect needs to be considered, too.

Betweenness centrality Betweenness centrality calculates the proportion of the number of shortest paths through v to the total number of shortest paths between two nodes—this is done for each node pair and summed up. Consequently, if there are several shortest paths, it is assumed that the flow process can only take one of them, and not several simultaneously as would be possible for duplication processes. This implies that the measure expects a process with a transfer mechanism [Bor05]. Additionally, by counting shortest paths, it is assumed that the flow process only travels along shortest paths. Similarly to the assumptions of the closeness centrality, this implies that each process entity has a target to reach, and knows how to reach it on a shortest path. Furthermore—also similarly to closeness centrality—it is assumed that there is potential traffic between each node pair and the traffic between each node pair is of equal importance. This is only true if there is a process flowing between each node pair—or if the probability that there is traffic between two nodes is equal for every node pair and betweenness centrality measures the *expected* amount of flow in which a node is involved.

Availability of edges Common to all introduced measures and generally to all path-based measures is the assumption that all paths in the graph are available. In static networks, edges that have been inserted once, are constantly present. This is different for dynamic networks where edges can be added or deleted over time, or temporal networks where each edge has timestamps indicating at which time points the edge is valid [HS12]. However, most network measures are designed for static networks in which any timestamps of edges are ignored. This is not always a valid assumption: In some systems, such as air transportation systems, connections are not always available, but only at certain time points. In other systems, there exist dependencies: a certain connection (b, c) might only be available for a process entity if the connection used directly before was (a, b) . This is, however, mainly a matter of representation. Consider, for example, networks constructed from trajectory data: Trajectories $a \rightarrow b \rightarrow c$ and $d \rightarrow b \rightarrow e$ will usually yield a network where the nodes a, c, d , and e are connected to node b . Aggregating the trajectories to such a network hides any dependencies contained in the trajectories. In this network representation, it is no longer taken into account that the edge (b, c) is only viable when coming from a . An alternative form of representation are so-called networks of higher order in which nodes do not represent single actors, but tuples of actors (for example (a, b)) [XWC16]. It is, however, not clear which order is sufficient: Second-order networks incorporate dependencies of two consecutive steps, third-order networks take into account dependencies of depth three, etc. Scholtes presents a model selection approach for determining the order of a network [Sch17]. For the first case, where edges have a timestamp, the usual approach when centrality measures are applied, is to ignore the timestamps and apply the centrality measure to the network without the timestamp. This is not unproblematic since the centrality values will be based on process flows which are not possible in a real system due to the non-availability of edges. For this reason, Scholtes et al. propose variants of centrality measures that only incorporate *time-respecting* paths, i.e., paths in which the order of the contained edges respects the order of their timestamps [SWG16].

2.2.2 Further properties of network processes

While Borgatti used the dimensions of trajectory type and mechanism of node-to-node transmission for categorizing network processes, we argue that network processes can be categorized by more than these two dimensions (see Table 2.2 for an overview):

Target to reach Network processes that move towards a goal will show different properties than those without a goal: In particular, they will stop when the target is reached while those without a goal will continue moving through the network. Based on other constraints, such as knowledge of the network’s structure, and the ability of the process entity to optimize its way through the network, the chosen paths might come close to the shortest paths. Network processes without a goal or with short-term goals like money transfer will look more like random walks on the infrastructure. Processes with a target can be further distinguished

into those where all entities have exactly the same target (a player trying to solve a game), and those where each entity has its own target.

Routing mechanism How is the trajectory of a process entity determined? There are processes such as a parcel being delivered, where the trajectory is predefined before the entity starts its trajectory. Hence, while moving through the network, no routing choices are made. This also requires global knowledge of the network structure to be available for determining the route (which we call *global routing*). This can be different for other processes. For a person browsing the Internet searching for some specific information, the global knowledge of the network structure is not available, but the decision on which page to visit next is made by the browsing person on the basis of the local view of the network (we call this scenario *local routing by process*). There is yet a third scenario, such as the propagation of a piece of information through a social network. The decision of which node the piece of information is forwarded to is also made on the basis of local knowledge of the network, but not by the process entity, but rather by the network node (*local routing by node*).

Forwarding or Adopting In the case of processes routed locally by the nodes, we can distinguish processes by the criterion of whether the *sending* node or the *receiving* node is of relevance: Consider the propagation of a gossip story where a person decides to whom the story is told. The propagation of a behavior or a trend is different: Although the sending node decides about showing the behavior or promoting a trend, it is actually the receiving node that decides whether to adopt it or not. For the latter, Centola distinguishes between *simple* and *complex contagion* [Cen10]: For simple contagion, the exposure to the process by a single neighbor node is sufficient to adopt it, while for complex contagion processes, a node needs to be exposed to the specific behavior from several neighbor nodes before it adopts it (in social networks, this is known as social reinforcement). For both cases, forwarding and adoption, a further distinction can be made regarding the aspect whether the forwarding or adopting of a process is a voluntary act of the node: Telling a gossip story or adopting a behavior is a voluntary act of the concerned node, while for the spread of an infection, this happens without the active decision of the concerned node. In the first case, there can be individual differences in the adopting/forwarding behavior of the nodes—some nodes might adopt a process with fewer exposure events than others. De Domenico et al. write “Despite the fact that information spreading shares some general dynamical features with the spreading of diseases, their nature is deeply different. [...] Information [...] is only worth spreading or not and this decision is made by individuals, unlike the case of disease spreading.” [DDLMM13] In the context of information diffusion, Milli et al. differentiate between active and passive diffusion [MRPG18]: The nodes of a network can be active in the sense that they can “decide” whether they want to adopt a behavior or another process. In other types of diffusion processes such as the spreading of a disease, the nodes are passive in the sense that the diffusion process will spread to them without them having any possibility to avoid it.

Interaction of network and process Do the process and the system interact in any way such that the one can change the other? In most cases, the network can be seen as the infrastructure in which the process can disseminate; thus, the process is restricted by the network structure. In other cases, there is a feedback loop such that, in the long run, the process can change the network structure. This is, for example, the case for transportation networks: When a connection between two airports is in high demand, but only an indirect connection via an intermediate stop is available, the airline might install a new direct connection. Hence, this would yield a new edge in the network representation. On the other hand, when a connection between two airports is not demanded by a sufficient number of passengers, the connection might no longer be offered by the airline; hence, the edge would be removed from the network. This effect has been analyzed in the context of information flow in social networks [WRP⁺13]. A further example is the spreading of diseases: If a disease is lethal, its spreading might lead to the removal of nodes of the network. Thus, the network process will

Table 2.2 Catalog of network process properties that determine the way an entity uses the network structure. The first two properties, transmission mechanism and trajectory type, were proposed by Borgatti [Bor05].

Dimension	Possible values
Transmission mechanism [Bor05]	transfer serial duplication parallel duplication
Trajectory type [Bor05]	shortest paths paths trails walks
Target to reach	same target for all entities individual targets no target
Routing mechanism	local routing by node local routing by process global routing
Adopting	forwarding adoption: simple contagion adoption: complex contagion
Interaction network and process	no interaction process shapes network

considerably change the network's structure by the removal of a proportion of the nodes.

2.3 Empirical analysis of network flows

As described in the previous section, the most frequently applied centrality measures contain a process model that uses shortest paths or a parallel duplication mechanism [Bor05] (see also Table 2.1). This insight raises the question for which real-world network flows the existing centrality measures are appropriate. In this thesis, we will consider datasets containing real-world network flows in order to test to which extent they fulfill the assumptions of centrality measures (in Chapter 3). Therefore, the following section reviews prior works analyzing the empirical properties of real-world network flows.

2.3.1 Empirical analysis of flows with a transfer transmission

For transfer processes, all introduced centrality measures only consider shortest paths (their existence, their length, or their number) [Bor05], apparently assuming that the relevant network process is moving on shortest paths. This also implies that the single process entities have a target to reach and that they know how to reach this target. While this might be true for several network processes, it has been shown for various processes that the aim to use shortest paths is rarely achieved overall.

Processes with a target to reach might involve humans—either as process entities themselves navigating through a network, or as nodes in a network forwarding process items. An example of the latter are social networks of humans who pass a piece of information to their acquaintances. While a piece of information normally does not have a target to reach, this is different in the famous experiment of Milgram from the late 1960s [Mil67]. Milgram asked randomly¹ selected people to

¹To be precise, participants were recruited by an advertisement in a newspaper specifically looking for persons who considered themselves as "well-connected".

send a letter to a target person by forwarding it to one of their acquaintances—who should then repeat this procedure until the letter eventually reached its target person. The experiment is also called *Small-world-experiment* since Milgram found that those letters that arrived at the target person only needed five intermediate steps, so this was surprisingly short. Although the experiment was not based on a large data base (in the different runs of the experiment, only 15 to 35 % of all letters actually reached the target person), it still gives a hint that humans are able to find surprisingly short paths in a network of which they only have a local view. This type of experiment was later repeated on a larger scale, for example by Dodds [Dod03] Adamic and Adar [AA05], and Backstrom [BBR⁺12]: Dodds recruited almost 100 000 participants who were asked to forward an email to a social acquaintance they considered to be closer to the target person (one of 18 target persons). He found similar effects: The majority of message chains did not reach the target persons, while the successful chains were quite short—only needing between five to seven steps.

For processes where humans navigate through a virtual or physical environment, it has been shown in various cases that humans are surprisingly efficient in finding a short way through the environment although rarely the optimal one. This observation has been made for human navigation in information networks, such as the network consisting of Wikipedia articles and the hyperlinks pointing to articles: West and Leskovec analyzed more than 50 000 human paths through this network collected by the game *Wikispeedia* where a player needs to navigate from a source to a target article [WL12b]. They found that the paths taken by the humans are rarely optimal, but on average only one step longer than the optimal path (with the optimal paths being between two and nine steps long). Similar results exist for human navigation in word networks where two words are connected by an edge if they differ in exactly one letter. Studies working on different datasets have found that the paths taken by humans are on average 1.7 times longer than the optimal path [SIVMZN12], when only experienced participants are considered, this factor even decreases to 1.1 to 1.2 [GBR⁺20].

Even for human navigation in physical environments where a global view of the environment is possible via maps, it has been shown that humans use short paths, but no shortest paths. This has been shown for human travel patterns within cities [ZL15] and for the routes of minicab taxis in London [MAC15]. It seems that the drivers prefer anchor-based routes, i.e., they use certain locations as landmarks for constructing the individual route, then first navigate to the landmark and from there to the destination.

For transfer processes where the navigation is not done by humans and where optimal paths could be expected, there is also evidence that this assumption is not necessarily true. Gao and Wang [GW02] analyzed the routes of packages being routed on the Internet. They found that more than 20 % of all considered paths were at least one hop longer than the shortest path. Csoma et al. [CKR⁺17] looked at trajectories of empirical network flows from different domains, including trajectories within the human brain, and also found a considerable number of non-optimal paths.

2.3.2 Empirical analysis of flows with a duplication transmission

For processes with a duplication transmission mechanism, there is a wealth of studies analyzing the properties of real-world network flows empirically. Since this thesis focuses on network processes with a transfer transmission mechanism, we will only name a few studies about duplication processes. Several studies provide details of information spreading on the Twitter platform, for example studies by Romero et al. [RMK11] or De Domenico et al. [DDLMM13], on the Facebook platform [BRMA12] or via email communication [IM09]. The spreading of memes has also been the subject of research [ALF12]. An overview of studies on information diffusion in online social networks is provided by Guille et al. [GHFZ13]. Christakis et al. [CF08, CF07] empirically analyzed the spreading of behavioral patterns or habits in a social network, such as obesity or smoking habits. They found that the behavioral patterns of close friends and family members have a signifi-

cant impact on the person’s own behavior. Investigating the spreading of a disease in networks is a whole research field in complex network analysis (and other fields): Pastor-Satorras and Vespignani studied the spreading dynamics of a computer virus in scale-free networks, Rocha et al. studied the spreading of infections in sexual contact networks [RLH11], and Salathe and Jones investigated the dynamics of infections in networks with a community structure [SJ10], to name but a few.

2.3.3 Using empirical network flow data for inferring knowledge about the system

Empirical real-world network flows show a different behavior than could be expected from the network structure. Therefore, it is possible to use real-world flows to infer additional knowledge about the system that could not be derived using only the network structure. This has been done in a few approaches. West et al. [WPP09] used human navigation trajectories through an information network to infer a semantic similarity between Wikipedia articles. Rosvall et al. [REL⁺14] used the data of real-world network trajectories to deduce community structures of the network. Weng et al. [WRP⁺13] used a dataset containing the actual diffusion of information in a social network, to predict the evolution of the network, i.e., the formation of new links. Yuan et al. [YZZ⁺10] made use of real taxi trajectories to compute the quickest path between places in a city: Thus, instead of using the global knowledge about the network structure, they used the collective knowledge of the taxi drivers contained in their trajectories to compute the effectively shortest path. A different approach is presented by Zheng et al. [ZZXM09] who used GPS trajectories of travelers to identify popular places. In Chapter 6, we will use the empirical human trajectories in a game’s state space to infer knowledge about cognitive process of human problem-solving.

2.3.4 Existing adaptations of centrality measures

The assumption a process is using shortest paths is not a realistic presumption in many cases. Stephenson and Zelen noted in a work in 1989: “It is quite possible that information will take a more circuitous route either by random communication or may be intentionally channeled through many intermediaries in order to “hide” or “shield” information in a way not captured by geodesic paths.” [SZ89] Thus, there exist adaptations of centrality measures containing shortest paths. These adaptations either relax the restriction of shortest paths by additionally allowing longer paths, or incorporate a different process model into the measure. An example for the incorporation of a different process model is Freeman’s flow betweenness centrality [FBW91] where not the number of *shortest* paths is counted, but rather the number of paths in which a node is contained in if there is a maximal flow between all node pairs. For the flow betweenness centrality, non-shortest paths can also contribute to the centrality value of a node. While the assumption of shortest paths is left aside for this variant, it is still assumed that the flow does have a target to reach, namely to get from each node to any other node. This might not be a realistic assumption in many cases. Furthermore, it is assumed that the process uses *ideal paths*—not in the sense of length, but with respect to the maximal flow. For this reason, Newman suggests a random walk betweenness centrality [New05]. He notes that “in most cases a realistic betweenness centrality should include non-geodesic paths in addition to geodesic ones” [New05]. The idea of his random walk betweenness is to sum the probabilities that random walkers in the network starting and ending in two nodes will visit node v on their random walk. A similar approach was already used earlier by Bonacich who proposed a power centrality that measures the expected number of times that a random walker with a fixed probability of stopping at each step, visits a node, averaged over all possible starting points for this walk [Bon87]. Similar adaptations exist for closeness centrality where instead of the length of the shortest paths from all nodes to v , the expected number of steps is considered that random walkers starting in each node need to reach v ; for example Markov centrality [WS03], or random-walk centrality [NR04]. PageRank centrality [PBMW99] is another example of a centrality measure based on the model of a random walker.

However, random walks are also often not a realistic approximation for relevant network flows: Chierichetti et al. [CKRS12] for example examined the assumption that web users can be modeled using a Markov chain. Similarly, Meiss et al. [MMF⁺08] considered human clickstream data and compared a relevance ranking of the pages obtained from real user traffic with the ranking obtained by the PageRank measure. They found that, assuming a random walker, PageRank was not able to rank the pages with respect to the actual user flow. Newman [New05] notes “our random-walk betweenness and the shortest-path betweenness of Freeman [Fre77] are at opposite ends of a spectrum of possibilities, one end representing information that has no idea of where it is going and the other information that knows precisely where it is going. Some real-world situations may mimic these extremes while others, such as perhaps the small-world experiment, fall somewhere in between.” Thus, in Chapter 3, we will also test to which extent the considered real-world network flow trajectories can be approximated by variants of random walks. In Chapter 4, we will then evaluate the impact on the results of the centrality measures that the real-world network flows are neither well-approximated by shortest paths nor by random walks.

Properties of network flows in complex networks

Chapter outline

In this chapter, we are concerned with the properties of real-world network flows. Particularly for centrality measures, it is known that they implicitly assume a process flowing through the network and also assume certain properties of the network flow [Bor05]. Therefore, we collected datasets containing the trajectories of real-world network flows from different domains. In this chapter, we will use the datasets of real-world network flows to examine whether they fulfill those assumptions: It is tested whether real-world network flows fulfill the assumption of shortest paths and the assumption of equal amount of flow between each node pair. Then, it can be judged whether—for example—betweenness centrality is a suitable measure for measuring the nodes' importance with respect to the actual network flow. We find that real-world network flows often deviate considerably from their assumed properties. This is why we will investigate whether other simple trajectory models, i.e., shortest paths and random walks, are able to reproduce the properties of real-world network flows. Since real-world network flows show different properties than these simple models, the next chapter (Chapter 4) will then investigate which impact these deviations have on the results of classic centrality measures.

The work in this chapter is mainly based on our publications [5] and [8].

3.1 Motivation

Most well-known centrality measures assume the presence of a network process with specific properties [Bor05] (see also Chapter 2). Consider the betweenness centrality as an example: To compute the betweenness centrality for a node v , we count for each pair (s, t) how many shortest paths from s to t contain v , in relation to the total number of shortest paths from s to t . These proportions of all (s, t) -pairs are summed up which yields the betweenness centrality for node v . The motivation for this measure is intuitive: If a node is positioned between many node pairs, this node is able to control the communication of these nodes and possesses a certain power over the other nodes by possibly *not* forwarding some information. This explanation already makes it clear that betweenness centrality is intended to measure a node's importance with respect to some network process, such as flow of information. When taking into account the formula, it can be seen that the process model presumed by this centrality measure needs to consist of indivisible items since one process entity can only use *one* path at once and not several simultaneously [Bor05]. Furthermore, whatever flows through the network, uses only shortest paths, since only these are counted. In other words, betweenness centrality implicitly incorporates a model of a network process consisting of indivisible items and using shortest paths. Furthermore, the number of shortest paths between each node pair contributes a value of at most 1 to the betweenness value, i.e., the amount of flow

from s to t is weighted equally for every node pair (s, t) . Thus, it is assumed that there is flow between each node pair, and the amount of flow between each node pair is equally important. Hence, a node with a high betweenness centrality value is only important with respect to processes with these properties. Using betweenness centrality to measure the importance of a node with respect to a process that does not have these properties, such as the spreading of a piece of information, will yield uninterpretable or even misleading results.

Borgatti also examined other well-known centrality measures for their incorporated process model and found that many relevant processes, such as infection or information spreading, have different properties than those assumed by most centrality measures [Bor05]. There are, however, network processes for which a transfer mechanism and the usage of shortest paths can be expected, for example for the flow of passengers in a transportation system. We therefore collected datasets containing information on how a network flow moves through a network and investigated to which extent the real-world trajectories are in accordance with their assumed properties. Thus, the main question of this chapter is:

Do real-world network flows satisfy the properties which are assumed of them by several network measures, particularly by centrality measures?

We therefore structured this chapter along the assumptions contained in centrality measures¹:

- (i) From where to where does it flow?
- (ii) How does the flow move from its source to its target?

For the first part, in Section 3.3, the real-world datasets are tested regarding the assumption that there is an equal amount of flow between each node pair. We find that this assumption is not fulfilled in any of the considered datasets: There are a few node pairs between which a large proportion of the total amount of flow is accumulated, while there is only a small amount of flow or even no flow between the other nodes. In Section 3.3.1, we also investigate whether standard network measures can explain the high usage of certain nodes. Section 3.3.2 provides an explanation on which node pairs are used more often than others.

For the second part, Section 3.4 contains an analysis of the available trajectories along the following questions (a) Which types of trajectories are found in real-world flow trajectories? (b) Is the assumption of shortest paths fulfilled? This also includes an analysis in Section 3.4.3 of whether all possible alternative paths are taken with equal intensity. The motivation for this analysis can again be best understood by looking at the example of betweenness centrality: If there are two shortest paths from node s to node t , and a node v is contained in one of them, this node pair contributes a value of $1/2$ to the betweenness value of v . This contains the assumption that either the flow from s to t is split equally among the two alternative paths, or the probability that the flow takes either of the alternatives is equal for both. In Section 3.4.3, we will therefore investigate whether this assumption is met by real-world network flows. As a start, a description of the datasets we used in our investigation will be provided in Section 3.2.

For all considered assumptions, Sections 3.3 and 3.4 reveal that the real-world network flows deviate from the properties of the process model contained in the network measures, particularly in centrality measures. One immediate question at this point is whether a different model than the one using shortest paths would be able to explain the properties of the real-world network flow. This is relevant since data on real-world network flows is not available in many cases. Thus, a

¹The assumption of being a transfer or duplication process cannot be tested, but is a matter of the choice of the datasets. Since most well-known centrality measures assume a transfer process, we restrict our analysis to datasets containing trajectories of transfer processes.

model that is able to reproduce the essential properties of real-world network flows could be used as a more realistic model. Apart from the model of shortest paths, another generic model for a network flow is based on random walks. Centrality measures such as PageRank [PBMW99], or random walk betweenness centrality [New05], the community detection method WalkTrap [PL05] or community detection by random walks [RB08], and link prediction by random walks [BL11] are examples of network measures incorporating a model based on random walks. In Section 3.5, we will therefore test whether several variants of a random walk model are a good proxy for real-world network flows.

3.2 Datasets used

In the following section, the datasets containing a network flow that we used for our investigation will be described. For the subsequent analysis, only network flows consisting of trajectories are suited, so, the following requirements need to be fulfilled:

- (i) It is a transfer process, i.e., the network flow consists of several indivisible items or entities moving from node to node. Thus, each entity or item is at one node at one point of time. This excludes network processes such as infections or information. Otherwise, the concept of trajectories or shortest paths is not meaningful to apply for those kinds of processes.
- (ii) The network representation and the process of interest need to be in accordance: The edges need to represent the type of relationship that is essential for the dissemination of the process. Considering the spreading of a virus in a social network would not really be suitable, since it is not the relationship “knowing a person” that is essential for the contagion of a virus, but rather the relationship “being in physical proximity to a person”.
- (iii) The network flow usually traverses more than one edge in a row, i.e., the network flow generates trajectories in the graph, and not only a set of dyadic relations (thus, trajectories of length 1)². This excludes network processes such as emotional support or seeking advice in social networks because the aspect of transitivity is not given: If A supports B and B supports C, there is no flow of support from A to C.

We use four datasets that fulfill these requirements which are described in the following (see Figure 3.1 for an illustration of the datasets used in this chapter, and Table 3.1 for their basic statistics).

London Transport We use a dataset containing passenger journeys within the London Transport system. The data is provided by Transport of London [Tra17], the governmental authority responsible for public transport within the region of Greater London. Every year, they publish the Rolling Origin and Destination Survey [Tra17] which includes the passenger journeys of a sample of 5% of holders of Oyster cards, an electronic ticket, using the London Underground system, during one week in November (we used the data for 2017), using London Underground system. Passengers with an Oyster card are required to check in and check out at entry and exit stations as well as at stations where they change trains. This makes it possible to know for each passenger trip the start, the destination, and the stations where train changes occurred. It is, however, not known which connection the passenger took or how much time they spent on the platform.

We used the publicly available timetables for the London transport system to construct a multilayer network of the system and to reconstruct which connection a passenger (probably) took, as described in the following. In this network, each layer corresponds to one Underground line, e.g., the Central line or the Victoria line, and each node represents an Underground station. In layer X , there is an edge from node v to node w , if station w can be reached from station v via line X without changing trains. This does not yield a chain-like

²Note that this requirement allows a network flow to contain trajectories of length 1, but not exclusively.

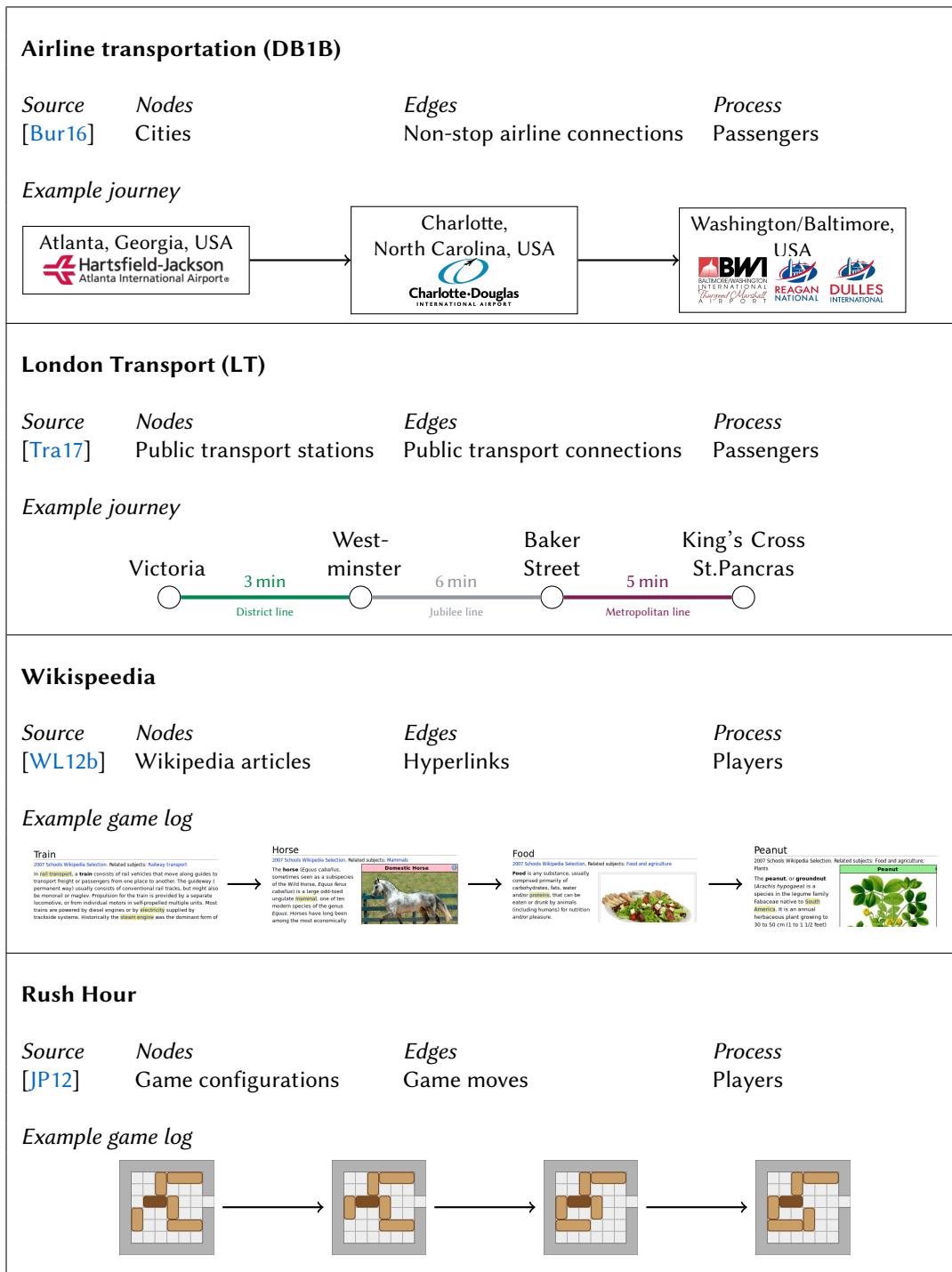


Figure 3.1 Overview of the datasets used in this chapter.

structure for each line as a route map of a line would suggest, but rather the transitive closure of the chain. The edges are weighted by travel time; more precisely, by the minimal travel time possible with the corresponding line. In addition to the multilayer network, we created a one-layer network by merging all layers into one. This yielded a network with 268 nodes and almost 14 000 edges. To extract the passenger journeys containing the stations where the passenger changed trains, we assumed that for each connection between two stations, the passenger took the line with minimal travel time. Note that this is a rough estimation and not necessarily true in all cases since the timing of the trains is not taken into account. It might be possible that the passenger actually took a different line since it left the platform earlier. However, since the precise times of the passenger journeys are not known, this estimation is the best possible. The complete trajectory is then the concatenation of these single connections. Note that in this modeling, a passenger's trajectory only contains the stations where the passenger changed trains, and not any intermediate stations where the train stopped, but the passenger did not get off. The reason for this modeling is the analysis in Section 3.5 where the observed trajectories are compared to random walks generated by an agent-based simulation. Such a representation of the trajectories and of the system makes a comparison of the random walks and the observed trajectories possible. The dataset contains more than 4.8 million passenger journeys, most of which were taken by a number of passengers which is why there are approximately 50 000 different journeys in the dataset. Due to inconsistencies in the data, 31 different journeys (a total of 37 000 journeys) were filtered out³.

Air transportation (DB1B) We used a dataset containing the passenger flow in the air transportation network in the US. The US Bureau of Transportation Statistics provides the Arline Origin and Destination Survey (DB1B) for every quarter year [Bur16]. This database contains the details of a 10 % sample of all airline tickets for flights within the United States (from all reporting airlines). We used the data published for the years 2010 and 2011 and extracted for each ticket the exact itinerary the passenger took, including origin and destination airport as well as all airports in which the passenger transferred to another aircraft. Most itineraries start and end at the same airport (passengers travel to their destination and then travel back home). Therefore, itineraries that include a "trip break" were split according to this attribute yielding two itineraries, an outbound and a return trip. We constructed a network where each node represents a *city* containing all airports of the city area (according to the database entry Market City ID); for example, the airports Chicago O'Hare International Airport and Chicago Midway International were both assigned to the city node of Chicago. An edge from node v to node w was inserted if the journey database contained at least one direct connection from an airport in v to an airport in w . This yielded a network with 462 nodes and almost 13 000 edges. For the extracted itineraries, basic consistency checks were performed: Itineraries not containing the same number of intermediate stops as indicated by the database were filtered out, as were itineraries containing the same airport as consecutive stops. In this way, we obtained more than 62 million passenger itineraries which took place within a period of two years.

Wikipedia As another dataset containing a network flow of indivisible entities with a target, we used human trajectories in information networks, provided by West and Leskovec [WL12b]. They considered human navigation in information networks such as the Wikipedia network, which consists of Wikipedia articles as nodes and hyperlinks between them as edges. West and Leskovec provide a publicly accessible, browser-based game on their website that interested users can play. In this game (Wikispeedia), the player is given a pair of articles (or chooses them on their own) and needs to navigate from the start to the target article by following the hyperlinks within the article. A public highscore is used in an attempt to motivate

³According to the official timetables provided by Transport of London, a connection contained in these journeys is not possible. It seems that this is due to changes in the route map executed after the data collection of the journeys and before we downloaded the official timetables.

Table 3.1 Basic properties of the datasets used. $|V|$ and $|E|$ denote the cardinality of the node and edge set of the underlying graph, $|\mathcal{P}|$ denotes the number of observed flow trajectories in the dataset. All networks consist of a single connected component.

Dataset	$ V $	$ E $	$ \mathcal{P} $	Trajectory length	
				Range	Average
London Transport	268	13 173	4.8m	[1, 107] min	16.3 min
DB1B	462	12 499	86m	[1, 12] hops	1.4 hops
Wikispeedia	4 589	119 804	51 306	[1, 404] hops	5 hops
Rush Hour Game A	364	1 524	3 044	[3, 33] hops	5 hops
Rush Hour Game B	6 769	33 142	1 965	[11, 59] hops	15 hops
Rush Hour Game C	830	4 037	1 472	[13, 95] hops	26 hops

the players to find the shortest possible path. The underlying network is not the complete set of Wikipedia articles, but rather a static subset of roughly 5 000 articles [Wik07]. West and Leskovec collected more than 70 000 solutions (and solution attempts) of almost 15 000 distinct players (more precisely, almost 15 000 distinct IP addresses). We used their dataset published on the SNAP network dataset collection [LK14] with the following adaptations: Due to a few missing links in the edgelist provided there [LK14] (which are however accessible when playing the game and therefore also included in the collected paths), we used the corrected edgelist that Robert West sent us upon request. The network consists of one giant component (containing almost all nodes of the network) and one isolated small component (containing three nodes). We limited our analysis to the giant component. Furthermore, we included only players' solutions that ended in their target article. While playing, the players were able to use an *Undo* button to withdraw their previous move (which they could also apply several times in a row). For our analysis, we excluded all moves that were revoked by using the *Undo* button.

Rush Hour As a second dataset containing game logs, we used a dataset containing human solution attempts for the single-player board game called Rush Hour⁴. In this game, the board with 6×6 cells and one designated exit represents a parking lot on which cars are placed. A player is given an initial configuration in which blocks (representing cars) are placed on the board. The player then needs to move the blocks such that a designated car (called target car) can be moved through the exit of the board. The blocks of width 1 and length 2 or 3 are placed horizontally or vertically on the board and can only be moved forward and backward, but not sideways. Figure 3.1 shows an example game configuration. A game configuration c induces a state space $G_c = (V_c, E_c)$ where a node represents a board configuration, and an edge represents a move transforming one board configuration into another. V_c thus contains the following nodes: one node representing the configuration c (the start configuration) and nodes representing all configurations that can be reached from c by allowed moves. A configuration is called a goal configuration if the target car can be moved through the exit. The set of goal configurations is denoted by $F \subseteq V_c$. We do not include nodes into V_c which can only be reached from the start configuration via a goal configuration. An attempt of a player to solve the puzzle is then a walk through the state space, starting in the node representing the start configuration and—if the puzzle was solved successfully—ending in a node representing a goal configuration. We thus distinguish between *solving* and *non-solving trajectories*. We used the trajectories of players for three games, collected by Pelánek and Jarušek [JP12] via a web-based learning platform. We only included solving logs, i.e., successful solution attempts. Table 3.1 shows the sizes of the corresponding state spaces and the number of available game logs.

⁴The game was invented by Nob Yoshigahara and is distributed by Thinkfun Inc. and HCM Kinzel (in Germany).

These datasets enabled us to investigate to which extent real-world network flows are in accordance with the process model contained in several network measures, which will be done in the following sections.

3.3 From where to where does it flow?

In this section, we will look at how the real-world network flows described above use the underlying network. Several network measures assume an equal amount of flow between each node pair. In the following, it will be tested whether this is a valid assumption for the available datasets.

3.3.1 Are all nodes equally likely to be contained in a trajectory?

Before the amount of flow between node *pairs* is considered, it is tested whether all network nodes are used by an approximately equal number of trajectories. We start by considering whether the real-world network flow uses the whole network or only a subset of nodes.

Are all nodes contained in at least one trajectory? We investigate whether the real-world network processes use the network in a uniform way, i.e., whether all nodes and edges are visited approximately equally often by the process trajectories. For this reason, we first consider the network coverage by the network flow:

Definition 3.1: Network coverage by \mathcal{P}

If \mathcal{P} denotes the set of real-world flow trajectories, we define the *network coverage by \mathcal{P}* as

$$C(\mathcal{P}) = \frac{1}{|V|} |\{v \in V | \exists P \in \mathcal{P} : v \in P\}|, \quad (3.1)$$

i.e., the proportion of nodes contained in at least one trajectory.

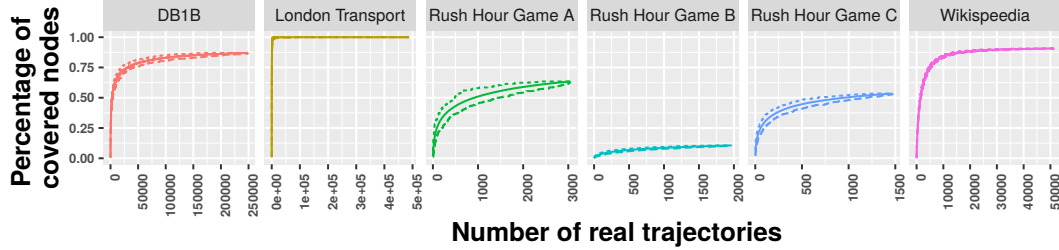
The table in Figure 3.2a shows the network coverage by the real network flow for each of the datasets. Due to the network construction of the DB1B and the London Transport network, it is clear that these networks show a coverage of 100%. For the game datasets, between 10% (for Wikipedia) and 89% (for Rush Hour Game C) of all nodes are not contained in any trajectory.

Is this finding due to insufficient data? The question arises whether the finding that not all nodes are contained in some trajectory is due to insufficient data or due to the nature of the flow process. In other words: if more data on the network flow was available, would the coverage values approach 100% or would parts of the network stay uncovered also with more data available? Note that the networks of all considered datasets are connected.

In order to answer this question, we perform the following experiment: For each dataset, we used the set of trajectories \mathcal{P} , successively picking (and removing) a trajectory P from \mathcal{P} uniformly at random, and add P to an initially empty set \mathcal{P}' . In every iteration, the coverage $C(\mathcal{P}')$ was computed. This procedure yields a sequence of increasing coverage values $(C(\mathcal{P}'_1), C(\mathcal{P}'_2), \dots)$ where \mathcal{P}'_i is the set \mathcal{P}' in iteration i . In order to reduce the effect of the order in which the trajectories are drawn from \mathcal{P} , this procedure was repeated $N = 500$ times, and the minimum, average, and maximum value of each $C(\mathcal{P}'_i)$ over all $N = 500$ iterations was computed. For the datasets DB1B and London Transport, where \mathcal{P} contains more than 63 million and almost 5 million trajectories, respectively, for computational reasons, not the complete set \mathcal{P} is used for the above procedure, but a randomly sampled subset (0.1% and 10% of the real trajectories, respectively). Figure 3.2b

Network coverage by \mathcal{P}					
DB1B	London Transport	RH Game A	RH Game B	RH Game C	Wiki
100 %	100 %	63 %	11 %	53 %	90 %

(a) Percentage of nodes covered by the network flow.



(b) Network coverage by the real-world network flows with increasing number of trajectories. Note that for DB1B and London Transport, not all real trajectories were used, but a randomly sampled subset.

Figure 3.2 Network coverage by the real-world network flow trajectories.

shows for each dataset the minimum, average, maximal coverage values for increasing number of trajectories in \mathcal{P}' . It can be seen that the network coverage of London Transport and Wikipedia reaches saturation with a small proportion of the available trajectories. While for London Transport, the coverage reaches the maximal possible value of 100 %, the coverage by the Wikipedia trajectories reaches saturation at 90 % and it does not appear that adding further real trajectories would increase the coverage. This also holds for the DB1B dataset; for the Rush Hour games, the coverage values increase more slowly with increasing numbers of trajectories, but no saturation could be verified. We conclude that for most of the considered datasets, the incomplete coverage of the networks is rather due to the nature of the process than to insufficient available trajectory data. If the coverage values increase with the same factor if further trajectories are added to \mathcal{P}' , a large number of trajectories would be needed to reach a coverage of 100 %. Note that for DB1B the coverage by the sampled subset of trajectories approaches approximately 90 % while the coverage by the complete trajectory set \mathcal{P} is 100 %. This already indicates that there might be an imbalance in the usage of the nodes by the trajectories: There appears to be a subset of nodes that are only contained in very few trajectories. Sampling a subset of trajectories, these nodes were found not to be contained in any of the sampled trajectories with high probability. The next section will investigate this imbalance of node usage more closely.

Are all nodes used equally often by the real-world network flow? In order to investigate whether the real-world network flow is present at all covered nodes with the same intensity, we consider the node usage:

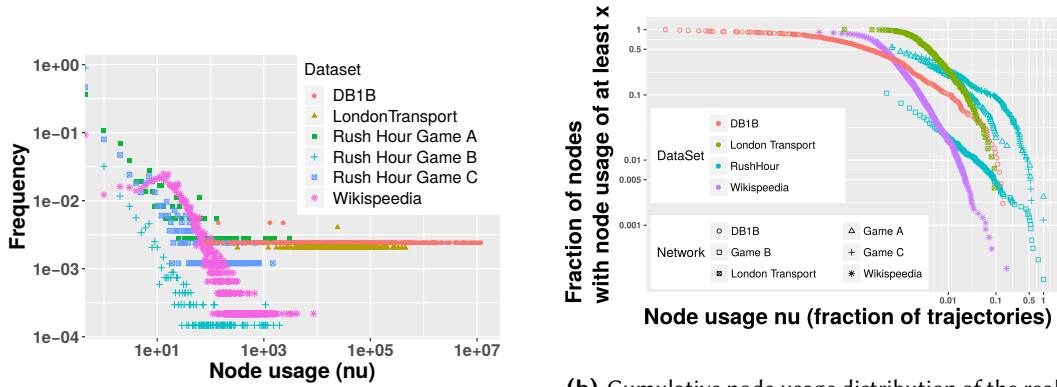
Definition 3.2: Node usage

We define the *node usage* of node v as

$$nu(v) = \frac{|\{P \in \mathcal{P} | v \in P\}|}{|\mathcal{P}|}, \quad (3.2)$$

i.e., the proportion of trajectories a node is contained in.

Figure 3.3 shows the node usage distribution for each dataset, Figure 3.3a shows for each node usage nu the frequency of nodes with this node usage (normalized by the total number of nodes



(a) Node usage distribution of the real-world network flows. In a double logarithmic representation, the node usage nu is on the x -axis, the proportion of nodes with a node usage $= x$ on the y -axis.

(b) Cumulative node usage distribution of the real-world network flows. In a double logarithmic representation, the node usage nu is on the x -axis, the percentage of nodes with a node usage $\geq x$ is on the y -axis.

Figure 3.3 Node usage distribution of the real-world network flows. Note the logarithmic scales in both plots.

in the network), Figure 3.3b shows the same distributions as cumulative distributions, i.e., for each node usage $nu(v) = x$, the y -axis shows the proportion of nodes with a node usage of at least x . It can be seen that for all datasets, the majority of nodes are only contained in very few trajectories—if they are contained in any trajectory at all—, while there are a few nodes that are contained in a large number of trajectories. Therefore, for all considered datasets, it is not true that the network flow process is present at all nodes with the same intensity. On the contrary: there is an imbalance in the node usage by the network flow.

Can network measures explain the high usage of certain nodes? Since it is observed that a few nodes are used heavily by the network flow while most nodes are used only once or not at all, the question arises whether network analytic measures can explain the high usage of certain nodes. We therefore computed several standard centrality measures for the networks at hand (see Chapter 2 for their definition) and compared the nodes' centrality values to their actual node usage. Figure 3.4 shows a graphic representation of this comparison. We furthermore computed Spearman's correlation coefficient [Spe04] of each centrality measure to the node usage (see Table 3.2). Only for the dataset DB1B did we find a high correlation between node usage and centrality measures (≈ 0.9). For the Wikipedia dataset, medium to high correlations (between 0.7 and 0.83) can be observed, while for the other datasets, the correlation coefficients are low or even negative for all centrality measures. This means that for the air transportation network, the actual node usage and the nodes' importance rated by standard centrality measures are in accordance—and this is true for all considered centrality measures. For the remaining datasets, including the other transportation dataset London Transport, standard centrality measures are not able to predict the actual network flow present in the nodes.

3.3.2 Are all node pairs equally likely to be source and destination of a trajectory?

We first consider whether all node pairs are used as source and target of at least one trajectory of the real-world network flows. The table in Figure 3.5a shows that for all datasets, there is no flow between the majority of the nodes. For the game datasets, this is not surprising: With 50 000 trajectories, the set of available trajectories for Wikipedia is simply not large enough to make it possible to observe a flow between all 21 million node pairs. For Rush Hour, all players start at the same start node and end (since only solving trajectories are considered) at one of the goal nodes.

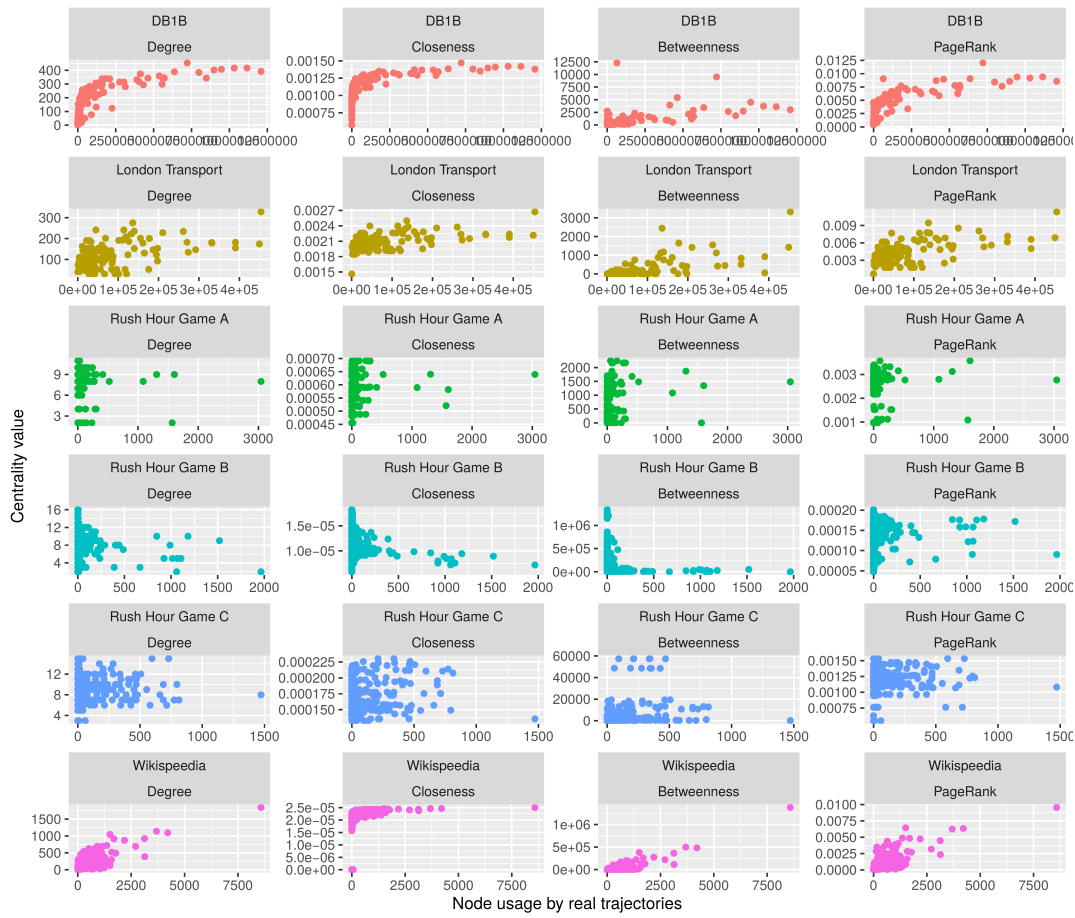


Figure 3.4 (Absolute) node usage by real-world network flows, compared to values of common centrality measures.

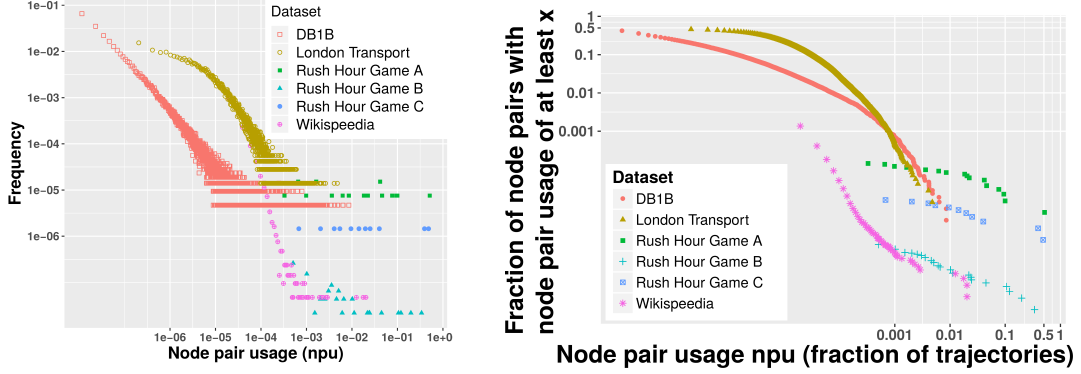
Table 3.2 Spearman’s correlation coefficients of classic network measures (node degree, closeness, betweenness [Fre77], and PageRank [PBMW99] centrality) to the actual node usage by real-world trajectories. Starred values indicate a p -value of < 0.05 .

Dataset	Degree	Closeness	Betweenness	PageRank
DB1B	0.96*	0.94*	0.89*	0.93*
LT	0.50*	0.51*	0.54*	0.54*
Wiki	0.71*	0.77*	0.72*	0.83*
RH Game A	0.007	0.15*	0.36*	0.16*
RH Game B	-0.15*	-0.29*	0.03*	0.01
RH Game C	-0.21*	-0.12*	0.29*	0.01

Percentage of node pairs used

DB1B	London Transport	RH Game A	RH Game B	RH Game C	Wikipedia
42 %	46 %	< 0.01 %	< 0.01 %	< 0.01 %	0.14 %

(a) Percentage of node pairs used as source and target of at least one trajectory.

(b) Source-target frequency distribution of real-world network flows. x -axis: logarithm of node pair usage npu of the node pairs; y -axis: logarithm of proportion of node pairs with a node pair usage = x .(c) Cumulative source-target-frequency distribution of real-world network flows. x -axis: logarithm of node pair usage npu of the node pairs; y -axis: logarithm of proportion of node pairs with node pair usage $\geq x$.**Figure 3.5** Source-target-frequency of real-world network flows.

If we consider only those node pairs (s, g) where s is a start and g a goal node, the percentage of node pairs used increases to 36 %, 3 %, and 11 %, respectively, which is still low. For the datasets DB1B and London Transport, both systems designed to bring passengers from one place to another, there is no network flow between more than half of all node pairs.

Similarly to the coverage and node usage in the previous section, we do not only consider the *number* of node pairs between which there is a network flow, but also the *intensity* of the flow between the node pairs. We therefore count for every node pair $(s, t) \in V \times V$ the number of trajectories starting in s and ending in t :

Definition 3.3: Node pair usage

We define the *node pair usage* npu for two nodes s, t as

$$npu(s, t) = \frac{|\{P \in \mathcal{P} | s(P) = s, t(P) = t\}|}{|\mathcal{P}|}, \quad (3.3)$$

i.e., the fraction of trajectories starting in s and ending in t .

Figure 3.5b and 3.5c show the distribution of the node pair usage of the node pairs for all datasets. On the left, the proportion of node pairs with a value of node pair usage is shown; on the right, the corresponding cumulative distribution is shown. It can be seen that for all datasets, the majority of node pairs are used only by one or two trajectories as source and target, while high values of node pair usage occur very rarely.

To illustrate how imbalanced the amount of flow between the node pairs is, we used a measure

that is often used in economics to show the income inequality of a population—the Gini coefficient (named after the Italian statistician Corrado Gini [Gin12, Gin21, CV12]). If a population’s income is distributed perfectly equally between all persons, each person has the same amount of income, i.e., here, if the amount of flow is distributed perfectly equally between all node pairs, there is an equal amount of flow between all node pairs. The previous section already showed that there are node pairs between which there is no flow at all which is why we focus on those node pairs between which there is at least one trajectory. An extremely unequal distribution then corresponds to the situation that almost the complete amount of flow is between one node pair, and only a minimal amount of flow is between the remaining nodes⁵. The Gini index is designed to quantify this degree of inequality. For the computation of the Gini coefficient, the Lorenz curve L is used where—in our case—the proportion of (used) node pairs is shown on the x -axis, and the proportion of flow is shown on the y -axis. A point (x, y) on the Lorenz curve then means that the $x\%$ least used node pairs accumulate $y\%$ of the total flow. Perfect equality will yield the identity line as a Lorenz curve; extreme inequality among the node pairs used will yield a Lorenz curve with y -values close to 0 for x between 0 and close to 1, and a steep connection to the point $(1, 1)$ at an x -value close to 1. Figure 3.6b shows the Lorenz curves for the source-target-distribution of the considered datasets where only those node pairs are considered between which there is at least one trajectory. The Gini coefficient is then the area between the Lorenz curve and the identity line (drawn in blue in Figure 3.6b), divided by the size of the area under the identity line.

The table in Figure 3.6a shows the corresponding Gini coefficients (the second row shows the resulting Gini coefficient if *all* node pairs are considered instead of only those between which there is at least one trajectory). Figure 3.6 shows the Lorenz curves and the Gini indices for all considered datasets. It can be seen that when all node pairs are considered, the Gini coefficient is very high or even close to 1 for all datasets. When only considering those node pairs between which there is flow, differences between the datasets can be seen. The DB1B dataset shows the greatest inequality in the distribution of the amount of flow between the node pairs. This is due to the fact that for example 75% of the total flow is among less than 2% of all (used) node pairs. Such an inequality has an effect on any analysis using centrality measures such as betweenness centrality: A node which is positioned between one of these high-demand node pairs should be rated as more important than a node which is positioned between the same *number* of node pairs, but which are used much less by the actual flow. However, this cannot be achieved by the betweenness centrality since it does not take into account the actual network flow, but each node pair is weighted equally. The most equal distribution (among the pairs used) is shown by the Wikispeedia dataset: Here, there are four node pairs with a larger amount of flow between them, while between 69% of all (used) node pairs, there is exactly one trajectory. The datasets London Transport, and the three Rush Hour games show similar values for the Gini coefficient, all above 0.73. This implies that even for the Rush Hour games where only those (s, t) -pairs are considered where s is a start node, and t is a target node used, there is a clear preference of the players regarding which target node is reached most frequently, while other target nodes are reached far less often.

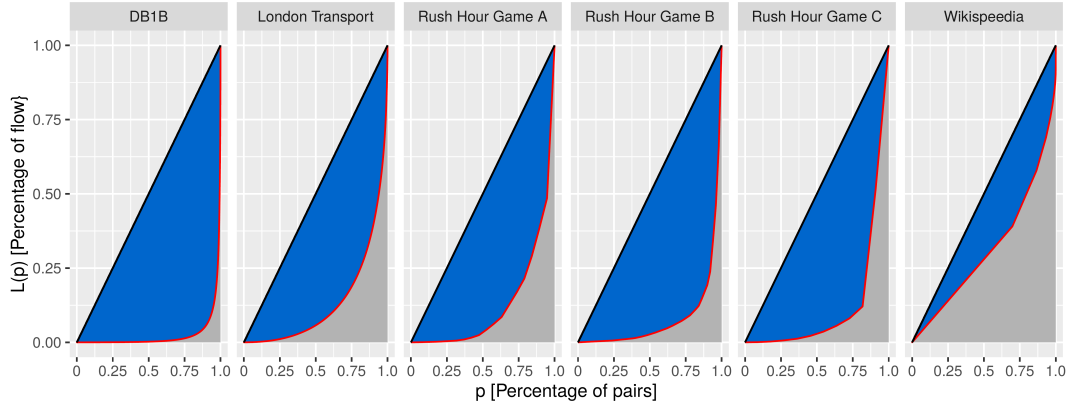
Which node pairs are used more often than others? In order to understand which node pairs are used more often as source and target than others (see also Table 3.3 for the five node pairs used most often by the trajectories in each dataset), we consider the graph distance of the node pairs. For that, we define a distance-based node pair usage:

⁵Originally, extreme inequality corresponds to the situation in which the complete amount of flow is between one node pair and no flow between the remaining node pairs. However, since only node pairs that were really used are considered here, this situation would actually represent a perfect equality

Gini coefficient for flow between (used) node pairs

	DB1B	London	RH Game A	RH Game B	RH Game C	Wikispeedia
Used pairs	0.95	0.73	0.74	0.84	0.75	0.38
All pairs	0.98	0.88	≈ 1	≈ 1	≈ 1	0.999

(a) Gini coefficient for the amount of flow between the node pairs used (first row) and between all node pairs (second row).



(b) Lorenz curves (drawn in red) for the distribution of flow between all node pairs with at least one trajectory between them. The Gini coefficient corresponds to the size of the blue area divided by the total size of the area underneath the identity line. Therefore, a Gini coefficient of 0 corresponds to perfect equality of the data (equal amount of flow between each node pair), while a Gini coefficient close to 1 corresponds to a high inequality of the data (large amount of flow between one node pair, and minimal amount of flow between the remaining node pairs).

Figure 3.6 The Gini index and the corresponding Lorenz curves for measuring the equality of the distribution of the total amount of flow among all (used) node pairs.

Definition 3.4: Distance-based node pair usage

For a graph distance k , it is counted how many trajectories start and end in node pairs with a graph distance of k . The node pair usage for a given distance k is thus defined as

$$npu(k) = |\{P \in \mathcal{P} \mid d(s(P), t(P)) = k\}| \quad (3.4)$$

Note that this definition does not take into account the trajectory length, but the *graph distance* of source and target node since the aim is to investigate whether a high amount of flow between node pairs depends on their distance. For this reason, it is not relevant which exact path is taken to get from one node to the other. In order to take into account that the total number of node pairs with distance k might vary for different values of k , we consider the total number of node pairs with distance k using

$$np(k) = |\{(v, w) \in V \times V \mid d(v, w) = k\}|. \quad (3.5)$$

For the London Transport dataset where the edges are weighted by travel time, we introduced distance intervals of 5 minutes such that a node pair with the distance $0 < k \leq 5$ is assigned the interval number 1, etc. For the DB1B dataset where an edge represents one flight connection, we additionally considered a weighted version of the network where an edge weight represents the geographic distance between the corresponding airports. For this network version, we also introduced distance intervals of 500 km.

Table 3.3 The five most frequently used source-target pairs for the datasets DB1B, London Transport, and Wikipedia. The second column gives the proportion of trajectories using the corresponding node pair as source and target.

Source-target pair	% \mathcal{P}	Source-target pair	% \mathcal{P}
San Francisco, CA → Los Angeles, CA	0.086 %	Asteroid → Viking	2.03 %
Los Angeles, CA → San Francisco, CA	0.085 %	Brain → Telephone	2.03 %
Miami, FL → New York, NY	0.064 %	Theatre → Zebra	1.76 %
New York, NY → Miami, FL	0.063 %	Pyramid → Bean	1.25 %
Los Angeles, CA → New York, NY	0.048 %	Batman → Wood	0.29 %

(a) DB1B

Source-target pair	% \mathcal{P}
Bank/Monument → Waterloo	0.48 %
Waterloo → Bank/Monument	0.41 %
King's Cross St. Pancras → Paddington	0.26 %
Victoria → Oxford Circus	0.24 %
Waterloo → Canary Wharf	0.21 %

(b) Wikipedia

(c) London Transport

Figure 3.7 shows for each dataset⁶ the number of trajectories starting and ending in node pairs of distance (or distance interval) k , normalized by the total number of node pairs with distance (interval) k . It can be seen that for all considered datasets, there is considerably more flow between node pairs that are closer to each other than between node pairs that are further apart. For the unweighted version of DB1B, the value of $n_{pu}(k)/np(k)$ is already close to 0 for $k = 2$. This is not too surprising since the network is often shaped by demand: If there is high demand between two airports, a direct flight between them will be installed and, vice versa, if there is only low demand between two airports, the direct connection might not be offered anymore. For London Transport, we found that particularly those node pairs that are close to each other are in much greater demand than node pairs with a greater distance. For the passenger flow within London, it can be expected that most passengers journeys are within the inner city center of London (where all stations are closer to each other) and not between peripheral stations, which can explain this result.

Although the result is not surprising for the datasets at hand, it does have consequences for network analysis: If a network measure expects a network flow in the network such as betweenness centrality, it is assumed that the probability of flow is equal for each node pair. This is, however, not the case for real-world network flows. A similar observation was made by Friedkin [Fri83] for communication networks: He found that in these networks, a distance threshold exists, which he calls the *horizon of observability*, beyond which the members of the network are not aware of each other. Thus, it is very unlikely to observe any flow between nodes that are further apart than this horizon of observability.

3.4 How does it flow?

While the previous sections considered the real-world network flows contained in the datasets with respect to the question of "from where to where does it flow?", the following analysis is concerned with the question of "how does the flow move from its source to its destination?". For this question,

⁶The Rush Hour dataset was excluded from this analysis since it is clear that all contained trajectories start at the same start node and end in one of the goal states.

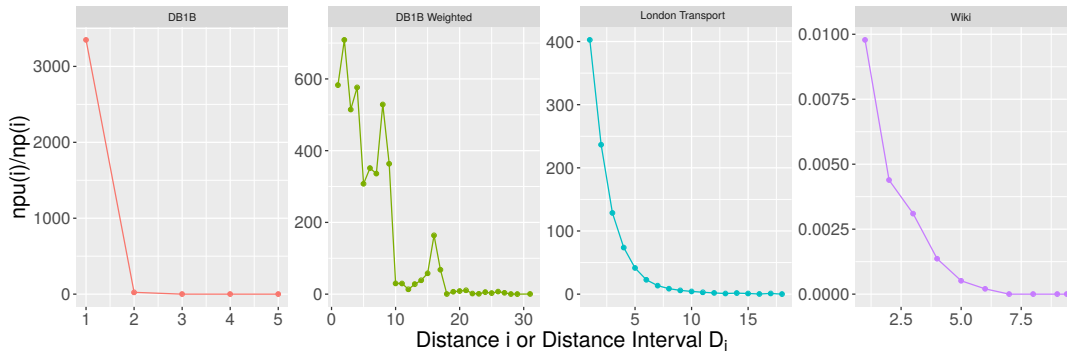


Figure 3.7 Source-target frequency by distance of the nodes. For each graph distance k , the total number of trajectories between nodes k steps away from each other, is normalized by the total number of node pairs in the graph with distance k . For datasets with weighted edges, the distance between nodes is divided into intervals of equal size: for London Transport, intervals of 5 minutes travel time, for the weighted network of DB1B with edges being weighted by the geographic distance between the airports, intervals of 500 km distance between the airports.

we first consider the types of trajectories contained in the datasets.

3.4.1 Trajectory types

Borgatti’s typology of network processes distinguishes flow trajectories by their type [Bor05]. He distinguishes between shortest paths, paths (trajectories containing each node and edge at most once), trails (trajectories containing each edge at most once), and walks (no such restriction). In order to be able to handle directed graphs, we extend this categorization: In directed graphs, it is possible that a trajectory contains both the edges $u \rightarrow v$ and $v \rightarrow u$ —which are different edges. A trajectory containing both the edges $u \rightarrow v$ and $v \rightarrow u$ is categorized as a trail although it contains the reverse edges $u \rightarrow v$ and $v \rightarrow u$. Thus, in order to differentiate between trajectories containing such reverse edges and trajectories not containing this pattern, we distinguish between four types of trajectories: (i) All nodes and edges in the trajectory are unique. (ii) At least one node is contained multiple times, but it does not contain any edge more than once and $(u, v) \in P \Rightarrow (v, u) \notin P$ for all $(u, v) \in E$. (iii) At least one node is contained multiple times in the trajectory, edges are unique, but there exists $(u, v) \in E$ with $(u, v) \in P$ with $(v, u) \in P$. (iv) At least one node and one edge is contained more than once in the trajectory. In Borgatti’s terminology, all four types fall under the category of a walk, types (i) to (iii) are called trails, type (i) is called a path. For trajectories of type (i), we furthermore checked whether it has optimal length, i.e., for a trajectory $P \in \mathcal{P}$, its length $|P|$ is equal to the length of the shortest path from the source to the target of the trajectories, i.e., $d(s(P), t(P))$.

For our used datasets, we computed the type of each trajectory. Table 3.4 shows the proportion of each trajectory type for our datasets. For the datasets containing transportation processes, we found that the proportion of trajectories of type (i) is close to 100%. Also for the Wikipedia trajectories, we observed a high percentage of this trajectory type, only 1% of the Wikipedia trajectories visited the same article more than once. This picture is different for the trajectories for the Rush Hour game instances. Even for the simple game A, 20% of the trajectories turned out to be of types (ii) to (iv) which means that at least one node was visited more than once. For game C, only 30% of the trajectories had unique nodes and edges. This is especially surprising since the dataset contains only solving paths.

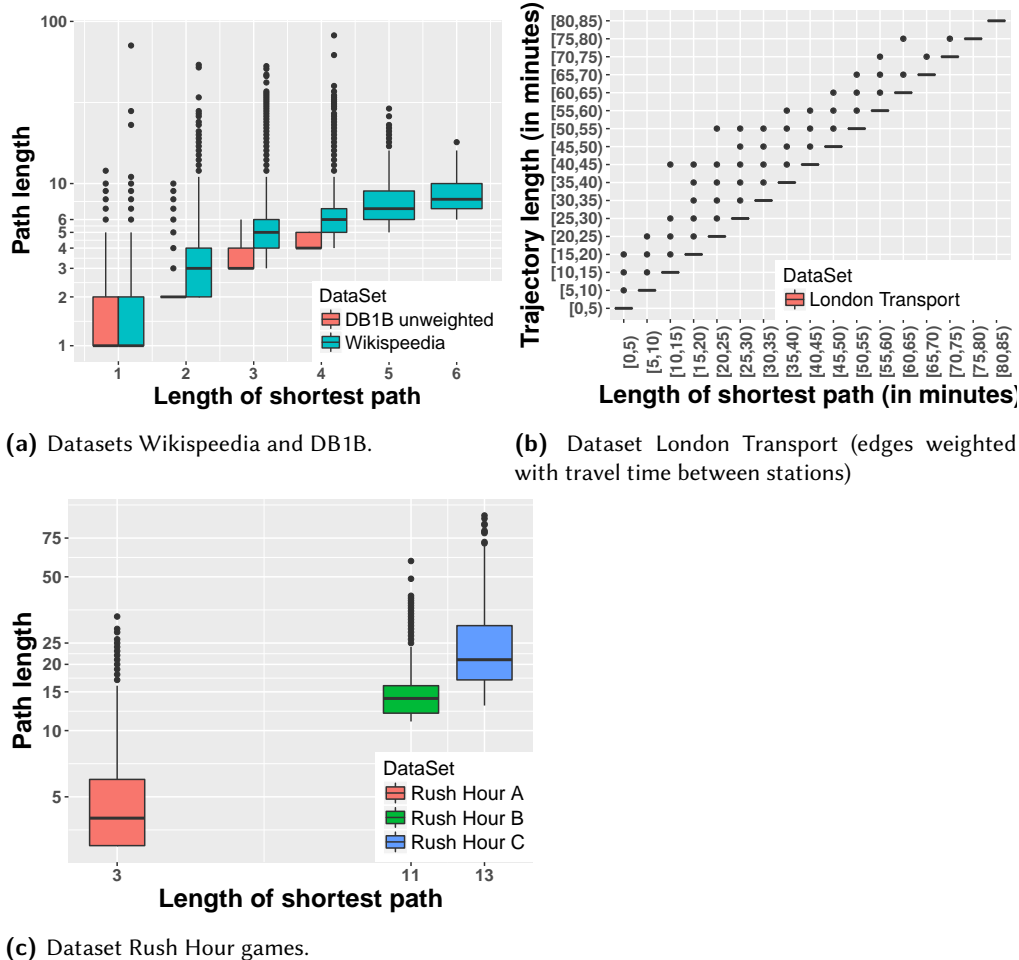
Table 3.4 Trajectory types of the real trajectories: Type (i) are paths in Borgatti’s terminology, types (ii) to (iv) are trails, type (iv) are walks in Borgatti’s terminology. Optimal denotes the proportion of trajectories with optimal length. The total number of trajectories is shown in the last column.

Dataset	Type (i)	optimal	Type (ii)	Type (iii)	Type (iv)	$ \mathcal{P} $
London Transport	99.7 %	89 %	0 %	0.3 %	0 %	4.8m
DB1B	99.8 %	69 %	≈ 0 %	0.2 %	≈ 0 %	86m
Wikispeedia	98.6 %	22 %	1.1 %	0.04 %	0.2 %	51306
Rush Hour A	78.8 %	37 %	0.8 %	18.5 %	1.9 %	3044
Rush Hour B	57 %	14 %	0.5 %	27.9 %	14.6 %	1965
Rush Hour C	30.8 %	1.8 %	1.8 %	36.9 %	30.5 %	1472

3.4.2 Is the network structure used optimally?

The previous paragraph showed that there is a non-negligible number of trajectories in the datasets that are longer than the corresponding shortest path. While this could be expected for the game datasets, it is surprising that even for the trajectories in the transportation systems, 11 % (London Transport), respectively 31 % (DB1B) of all trajectories were longer than needed. In the following, we will consider the lengths of the trajectories in order to investigate whether the assumption of shortest paths is a valid assumption for the considered real-world network flows. In Figure 3.8, the length of the trajectories is compared to the length of the shortest path from source to destination. Thus, for all trajectories in \mathcal{P} with a distance of x from source to destination, the length of the trajectories is shown as boxplot. For the Rush Hour games, all trajectories start at the same initial node and end at one of the goal nodes which is why the length of the shortest path is equal for all trajectories for each game. For DB1B, Figure 3.8a shows that for each length of the optimal path, the median of the length of the real trajectories is equal. Note that there is no trajectory where the source and the destination have a distance greater than 4. However, particularly for trajectories with an optimal path length of 1, there is a considerable number of real trajectories of length 2. Overall, the itineraries were on average longer by a factor of 1.3 than the optimal path in the network (counted in hops). For the other transportation system, London Transport, where the edges are weighted by travel time, the length of the real trajectories and the corresponding optimal paths are divided into intervals of 5 minutes. Figure 3.8b shows that the length of the real trajectories is very close to the length of the corresponding optimal path, on average, the travels are longer than the optimal path by a factor of 1.02. There are, however, outliers where the length of the real trajectories is longer than the optimal path by up to 30 minutes. As expected, for the game datasets, most real trajectories are longer than the corresponding optimal path. However, for Wikispeedia, the real trajectories are longer than the optimal path by only one step on average (see Figure 3.8a). For the three Rush Hour games (see Figure 3.8c), the lengths of the real trajectories strongly depend on the game instance: For the easy game A with an optimal solution length of three steps, the real trajectories are on average of length 5 (median 4); for game B with an optimal solution length of 11, an average length of 15 (median 14) is observed for the real trajectories; for game C with an optimal solution of 13, the average length of the real trajectories increases to 26 (median 21).

Thus, for the transportation processes, the real trajectories are close to optimal paths. There is, however, a considerable number of real trajectories that are longer. For the game datasets, the real trajectories are longer than the shortest path by—on average—1 for Wikispeedia and by up to 13 steps for Rush Hour.



(a) Datasets Wikispeedia and DB1B.

(b) Dataset London Transport (edges weighted with travel time between stations)

(c) Dataset Rush Hour games.

Figure 3.8 Lengths of trajectories for the datasets used, shown as boxplots. For Wikispeedia, three trajectories of length 404, 108, and 101 were removed to improve readability.

3.4.3 When several alternative paths are possible, are all possibilities used equally often?

Usually, for a process entity traveling from node s to node t in the network, there are several possibilities to reach t . This is often also true if the network flow only uses shortest paths. Since for network measures assuming a network flow, it is not known which route the network flow takes in order to reach t from s , it is assumed that each alternative route between s and t is taken with equal probability. For betweenness centrality, for example, consider two nodes s and t with two shortest paths between them. For each node on the two shortest paths from s to t , it is then assumed that it can control half of the flow between s and t .

For real-world network flows, it is not obvious whether all alternative routes are taken with equal probability, i.e., whether the amount of flow from s to t is distributed equally on all alternative routes. It is also possible that most flow is concentrated on one particular route while alternative routes are only rarely taken. This section describes the analysis we performed to investigate which case is true for real-world network flows.

Since the previous section showed that real-world network flows often do not use shortest paths, the following analysis cannot be limited to *shortest* paths between node pairs, but needs to consider

Table 3.5 The table shows the number of node pairs (s, t) that were used for the analysis of alternative routes (results shown in Figure 3.9). The left column contains the number of node pairs $(s, t) \in V \times V$ between which there is at least one flow trajectory in the dataset, the right column shows the number of node pairs between which there are at least 10 trajectories in the dataset. Only these node pairs are used for the analysis of alternative routes.

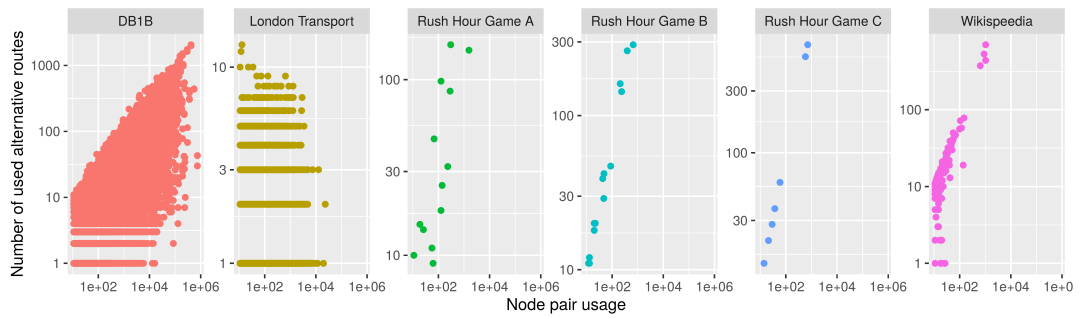
Dataset	$\#(s, t)$ with $npu(s, t) \geq 1$	$\#(s, t)$ with $npu(s, t) \geq 10$
DB1B	89 968	50 376
London Transport	32 905	26 982
Wikispeedia	28 706	160
Rush Hour Game A	19	13
Rush Hour Game B	50	17
Rush Hour Game C	11	7

all possible walks between node pairs. However, if the length of the walks is not restricted, there can be infinitely many walks between the nodes s and t , since cycles—if contained in the walk once—can be looped arbitrarily often. For this reason, we focus on the walks actually *taken* by the network flow.

We will first consider the number of different trajectories between node pairs before considering how the flow entities are distributed among the alternatives. For a node pair (s, t) and all real trajectories starting in s and ending in t , we count how many different trajectories are contained in the real-world trajectories where each of the different trajectories is also called an *alternative route*. Each alternative route is used by at least one flow entity in the dataset. Counting alternative routes between nodes is only meaningful if several flow entities travel between these nodes. Therefore, only node pairs (s, t) with at least 10 real trajectories from s to t will be considered in the following. Table 3.5 shows the total number of node pairs (s, t) with at least 1 and at least 10 trajectories from s to t . Excluding node pairs with less than 10 trajectories between them substantially reduced the number of considered node pairs. Nevertheless, this step is necessary for a meaningful analysis of route alternatives.

Figure 3.9a shows for each dataset and each node pair (with the restrictions described above), the number of alternative routes taken and its (unnormalized) node pair usage, i.e., the number of flow entities traveling from s to t . For Wikispeedia and also for the Rush Hour games, it can be seen that the number of alternative routes taken increases with increasing node pair usage, i.e., the more entities move from s to t , the more alternative routes are taken. This is not true for the transportation datasets. For London Transport, we found that only very few alternative routes were taken; most were between 1 and 10, independent of the node pair usage. For DB1B, on the other hand, for each node pair usage, there are almost all possible values of the number of alternatives taken. In the extreme case, there are more than 2 000 route alternatives (with a total of more than 418 000 trajectories between the corresponding nodes).

However, when we consider the total number of node pairs between which such a high number of alternative routes was taken, it turns out that these are only the extreme cases. The table in Figure 3.9b shows the proportion of node pairs between which exactly one route was taken by all flow entities. Note that here as well, only those node pairs are included between which there are at least ten trajectories. Figure 3.9c shows the corresponding histogram: For each number of route alternatives between a node pair, the number of node pairs with this value is shown. It can be observed that for all datasets, only a small number of route alternatives was taken for most node pairs. For London Transport, we even found that for more than 61 % of node pairs used, exactly one route was taken by all flow entities.

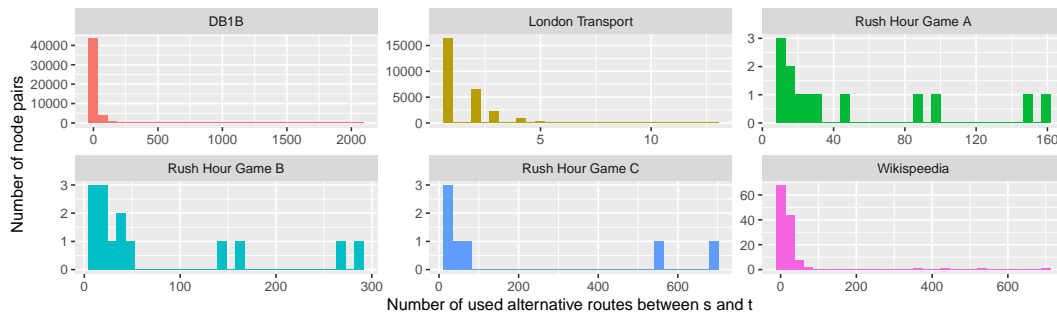


(a) Number of trajectories between a node pair versus number of different trajectories between the node pair. Each point represents one node pair. Note the logarithmic scales on both axes.

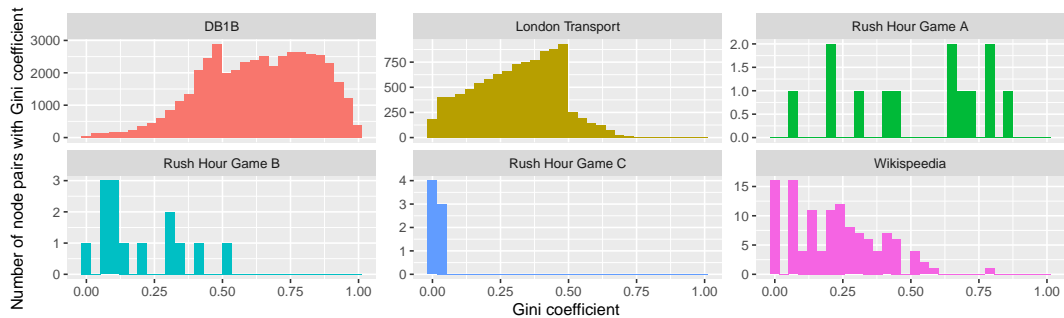
Percentage of node pairs where only one alternative is used

DB1B	London Transport	RH Game A	RH Game B	RH Game C	Wiki
8.0%	61.9%	0%	0%	0%	4.8%

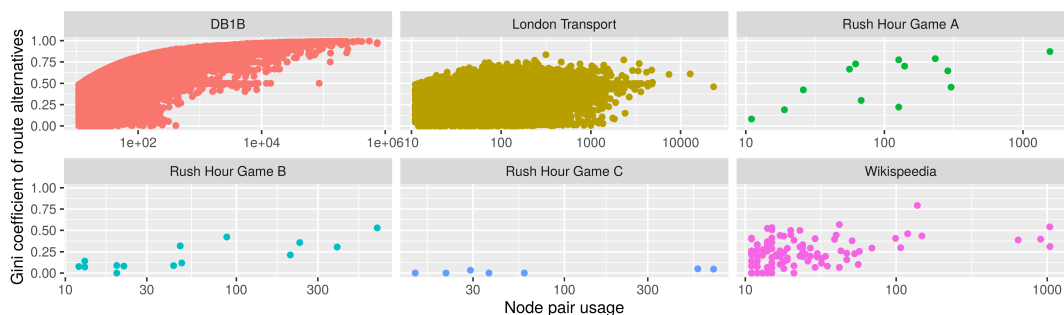
(b) Percentage of node pairs between which exactly one route is taken by all trajectories. Note that only those node pairs are included between which there are at least ten trajectories.



(c) For each node pair between which there are at least ten trajectories, we counted how many different alternative routes were taken. The histogram shows the number of node pairs with the corresponding number of alternative routes.



(d) For node pairs between which more than one alternative route was taken and between which there are at least ten trajectories, the Gini coefficient for the distribution of the trajectories among the alternative routes for each node pair was computed. The histogram shows the number of node pairs with the corresponding Gini coefficient value.



(e) Gini coefficient of a node pair versus number of trajectories between the node pair. Only node pairs with at least ten trajectories between them and where more than one possible route was taken are depicted.

Figure 3.9 Analysis regarding equal use of alternative routes between node pairs.

For those node pairs between which more than one route was taken by the real-world flow, we looked at whether the flow entities are distributed equally among all alternative routes or whether most entities used one route while the remaining alternative routes were taken only by a few entities. For this purpose, we used the Gini coefficient introduced in Section 3.3.2. For each node pair (s, t) (with at least ten trajectories and more than one route alternative from s to t), we counted how many flow entities used which route alternative from s and t . The Gini coefficient was then computed for this distribution of each node pair. A Gini coefficient close to 0 for a node pair (s, t) then means that all route alternatives were used by an approximately equal number of flow entities; a Gini coefficient close to 1 for a node pair means that one route alternative was taken by a large number of flow entities, and the remaining route alternatives were only used by one flow entity each. Figure 3.9d shows the frequency distribution of the computed Gini coefficients for each dataset. It shows that the situation is different for the considered datasets. For DB1B, in particular, higher Gini coefficients are observed, indicating that in most cases, one route alternative was clearly preferred. When we look at the data to see how these high Gini coefficients can be explained, we find that for many (s, t) -pairs with a graph distance of 1, the majority of the passengers took the direct connection from s to t . For those node pairs, the dataset also contains journeys of passengers who traveled from s to t via an indirect connection. The existence of these cases can be explained by various reasons—temporary unavailability of the direct connection, price-related advantages of the indirect connection, or private reasons for intermediate stops. Therefore, since the majority of passengers took the direct connection, but a (relatively) small number of passengers were distributed among these indirect connections, high values of the Gini coefficient are found.

For Wikipedia on the other hand, rather low values of the Gini coefficient are found, indicating that in most cases, the players were distributed almost equally among the route alternatives. But here as well, there are cases where one route was clearly preferred: For the Wikipedia trajectories from the node BIRD to the node GREAT WHITE SHARK, 90 of all 138 trajectories between them used the route over the nodes FISH→WHITE SHARK→SHARK, 29 used another alternative route—and the remaining 17 route alternatives were only used by one entity (and one alternative was used by three entities). Interestingly, this preferred route alternative is not the shortest path from BIRD to GREAT WHITE SHARK. The shortest path over the nodes PENGUIN→ORCA was not taken by a single player. However, in general, for most node pairs of Wikipedia, rather low values of the Gini coefficient are found, such as for the node pair (AFRICA, DIFFERENTIAL EQUATION) where each of the 13 alternative routes was taken by exactly one or two players.

For the London Transport dataset, almost no values of the Gini coefficient larger than 0.6 are observed, indicating that for this dataset, there are almost no node pairs between which there exists one preferred route. Most of the Gini coefficient values we found are between 0.25 and 0.5. However, it needs to be noted that for this dataset, less than 40% of all node pairs are included in this figure, because for the remaining node pairs, only one route was taken by all flow entities. For the node pairs between which more than one route was taken, Figure 3.9c shows that the flow entities only used two or three alternative routes in most cases.

In order to investigate whether only node pairs with a large amount of flow between them show high or low values of the Gini coefficient for the distribution of the route choice, the number of trajectories between a node pair was plotted against its Gini coefficient value, as depicted in Figure 3.9e. Also here, only node pairs with at least ten trajectories between them and between which more than one possible route was taken, were considered. It can be seen that such a relationship only exists for the DB1B dataset. Here, large values of the Gini coefficient, i.e., a clear preference for one route, is only present at node pairs with a large amount of flow. There are, however, also node pairs with large amounts of traffic and lower values of the Gini coefficient.

3.5 Can properties of trajectories be explained by a random walk model?

The previous sections showed that the real-world network flows show a different behavior than it is assumed by centrality measures where an equal amount of flow between each node pair is considered, possibly on shortest paths. For real-world network flows, we found that there is a large amount of flow between a few node pairs while almost no flow exists between the majority of node pairs. We found a similar effect for node usage: A few nodes are contained in many flow trajectories while most are only contained in a few trajectories. At the same time, we found that even for those network flows where each flow entity has a target to reach as fast as possible, there are a considerable number of trajectories that do not use the shortest path.

This means that network measures assuming an "ideal flow" such as some centrality measures, will yield inaccurate results if the real flow deviates from the "ideal" behavior (see Chapter 4 for detailed analysis of centrality measures assuming network flows). However, since datasets of real-world network flow are (i) rarely available, and (ii) highly dependent on the data collection methods and data preprocessing steps, the question arises whether a simple reproducible model may be able to produce a similar behavioral pattern as real-world network flows. If this is the case, datasets of real-world network flows are not needed: the process model can be used instead. The two simplest models for simulating network flow are shortest paths and random walks. In some sense, shortest paths and random walks are the extreme cases of the same scale as Newman describes it for the case of shortest path and random walk betweenness: They "are at opposite ends of a spectrum of possibilities, one end representing information that has no idea of where it is going and the other information that knows precisely where it is going" [New05].

Simulating network flows with a transfer transmission mechanism using models has been done in various contexts. Most are based on random walks or some variant of a Markov chain. Sen and Hansen [SH03] considered human browsing of the web within a single site and propose a model for predicting the user's next click, by using Markov models. Manley [Man15] presents a Markov chain simulation approach for modeling the traffic in London, by using the trajectory data of minicab routes in London. Rosvall et al. simulated real-world network flows, such as passenger flows in air transportation and taxi trajectories, also using a second-order Markov chain [REL⁺14]. The commonality of both approaches is that their process model is based on real-world network flows: the probability to which node the simulated flow moves next is not independent of its previous steps, but the transition probabilities are computed based on the real-world trajectories. West and Leskovec [WL12a] describe agent-based navigation in information networks where the agents' decisions are not based on the human trajectories, but rather on simple numerical attributes of the nodes. An approach in which several routing strategies are compared to each other is presented by Lee and Holme [LH12]. They considered spatial networks, i.e., networks in which each node is associated with a spatial position given in coordinates. They implemented a "greedy navigator", an agent-based exploration where each agent is equipped with some local sense of orientation, and compared the resulting trajectories with those of shortest-path-navigators and random walk navigators.

Section 3.4.2 already showed that the considered real-world network flows cannot be perfectly reproduced by shortest paths, and we also do not expect that they will be perfectly reproduced by random walks. These two possibilities are, however, the two extreme cases that determine the possible space for the flow behavior. In the following, we will investigate whether real-world network flows can be simulated using shortest paths or random walks to reproduce certain properties of their behavior.

3.5.1 Simulation using random walks

The most simple random walk model consists of a *random walker* which can be at one node at a time. In each step, the random walker uniformly at random chooses one of the neighbor nodes of the current node and moves to the chosen node. There exist many network measures that are based on random walks; for example centrality indices such as Page Rank [PBMW99] or random walk betweenness [New05], or community detection through random walks [PL05]. Most of them do not rely on simulations of a random walker, but can be solved analytically by considering the stationary distribution of the corresponding Markov chain in which the states represent the position of the random walker. The stationary distribution then yields for each node the probability that the random walker is at this node. For an infinitely long random walk with uniform probability of choosing a new neighbor and a probability of $1/|V|$ for each node v that the random walker starts its random walk in v , the stationary probability distribution is a function dependent on the nodes' degree (in undirected graphs) [MPL17].

In this work, we used a simulation approach for the following reason: The previous section showed that real-world network flows show a source-target frequency which is far from being uniform in the sense that the probability that a random walker starts in node s is equal for all nodes. We therefore need to take into account this skewed distribution in order to distinguish which behavioral pattern is due to the source-target distribution and which is due to the way the process entities move through the network. This is why we used the following simulation procedure in which the simulated random walks are tied to the set of real trajectories \mathcal{P} in the following way: for each $P \in \mathcal{P}$, a random walk is started in $s(P)$, chooses a neighbor node, moves to the chosen node, and continues this procedure until a stopping criterion dependent on the length of P is reached. In this procedure, the number of generated random walks and real trajectories is equal, their source-distribution is equal, as is their "outreach potential". This allows separating the two aspects that might impact the flow's behavioral patterns: (i) From where to where does it flow?⁷ (ii) How does it flow through the network?

The following variants of the basic procedure were implemented which differ with respect to the choice of the neighbor nodes and the stopping criterion:

Neighbor choice Two variants of neighbor choices were implemented:

Uniform neighbor choice (UNC) The simplest variant of choosing the next node to which the random walker moves next is to select a node uniformly at random from the set of all neighbors of the current node.

Backwards-restricted choice (BWR) Another variant for choosing the next node is to select uniformly at random from the set of neighbors of the current node where the node visited directly before is excluded.

Stopping criterion Both implemented stopping criteria were intended to ensure that the random walks had the same "outreach potential" as the real-world trajectories.

Path length restriction (PL) The random walker continues its random walks as long as the length of its walks does not exceed the length of the corresponding P . This yields a random walk Q_{rand} with $|Q_{rand}| = |P|$. For the case of weighted networks, the random walk is bounded by the weighted length of P . In this case, the random walker continues as long as $|Q_{rand}| \leq |P|$. If the random walker chooses an edge where adding its weight to the weight of the current random walk, would exceed $|P|$, this edge is not added to the random walk, and the random walker stops. This yields a random walk with $|Q_{rand}| \leq |P|$.

Line change restriction (LC) For the London Transport dataset, where the system is modeled as a multi-layer network with each layer corresponding to one train line, a random walker with path length restriction would yield a random walk with many layer

⁷In the described procedure, only the source distribution can be considered because the random walk does not necessarily end in $t(P)$.

changes. In the system, this corresponds to a journey with many line changes where the passenger gets off the train, possibly changes platform, and gets onto another train. This does not necessarily correspond to a passenger's behavior since in reality changing lines takes additional time (which is not included in the modeled trajectories—here, changing the layer does not increase the length of the trajectory). We therefore introduced a second stopping criterion particularly for random walks in the London Transport network where the random walker is restricted by the number of line changes contained in P . In each node, the random walker chooses a neighbor node. If this node can be reached within the current layer, the random walker moves to the chosen node and no change of lines is registered. If this node is only reachable by changing a line, the number of line changes made by the random walker is increased by 1. This procedure is continued until the random walker's number of line changes is equal to the number of line changes in P .

For each dataset and the corresponding trajectory set \mathcal{P} , sets of random walks tied to \mathcal{P} were computed. In the following, the properties of the generated random walks will be compared to the observed properties of the real trajectories.

Network coverage We first checked whether the generated random walks yielded a similar network coverage as the real trajectories (see also Figure 3.2). We therefore repeated the coverage experiment described in Section 3.3.1 with random walks. We also included shortest paths so that for network coverage with increasing number of real trajectories, random walks and shortest paths can be compared. Algorithm 3.1 illustrates the method we used: For each drawn real trajectory P , the corresponding random walk was generated and a shortest path from $s(P)$ to $t(P)$ was computed. The network coverage by the set of real trajectories, random walks, and shortest paths was computed. As in Section 3.3.1, this procedure was repeated 500 times in order to reduce the effect of the order in which the real trajectories were drawn, and the minimum, average, and maximum coverage by each subset was computed.

Figure 3.10 shows the network coverage by real trajectories, random walks, and shortest paths with increasing numbers of trajectories. For the datasets Rush Hour, Wikispeedia, and DB1B, only the results for random walks with uniform neighbor choice are shown, since the results for random walks with backwards-restricted neighbor choice are qualitatively the same.

The results are different for the different datasets. For DB1B (see Figure 3.10c), we found that with an increasing number of trajectories, the coverage coincides perfectly for real trajectories and shortest paths which is in accordance with the previous observation that 69 % of all real trajectories are shortest paths. While the coverage by real trajectories and shortest paths approaches a value of 90 %, the coverage by the corresponding random walks reaches a value of 100 %. Note that the complete set of real trajectories also reaches a network coverage of 100 %, however, for this experiment (as in Section 3.3.1), a randomly sampled subset of trajectories (0.1 % of all trajectories) was used. This indicates that there is a subset of airports that is used occasionally by flights (and is reachable for a random walker), but much less often than the remaining airports. This finding supports the previous findings, indicating that the real-world network flow is very different from the "ideal" flow as assumed by centrality measures.

For Wikispeedia, Figure 3.10b shows that the coverage pattern almost coincides for real trajectories, random walks and shortest paths. This is surprising since the analysis in Section 3.4.2 showed that real trajectories are not shortest paths, but only longer than shortest paths by one step on average. Thus, real trajectories are neither shortest paths nor random walks, but still show the same coverage behavior.

For London Transport (see Figure 3.10d), all types of trajectories—real, random, and shortest—reach a coverage of almost 100 % very fast. Note that also here (as in Section 3.3.1) a randomly sampled

Algorithm 3.1: Procedure of computing the coverage values by the observed trajectories, shortest paths, and random walks. After termination, 500 sequences of coverage values (each) for observed trajectories, by shortest paths, and by random walks are saved which can then be used for further analysis.

```

Data: graph  $G = (V, E)$ , set of walks  $\mathcal{P}$  in  $G$ 
1 for 500 times do
2   Initialization:
3   Sequence of coverage values by observed trajectories:  $C = ()$ 
4   Sequence of coverage values by shortest paths:  $C_{sp} = ()$ 
5   Sequence of coverage values by random walks:  $C_{rand} = ()$ 
6    $\mathcal{P}' = \emptyset$ 
7    $\mathcal{P}_{RAND} = \emptyset$ 
8    $\mathcal{P}_{SP} = \emptyset$ 
9   while  $\mathcal{P} \neq \emptyset$  do
10    draw (and remove)  $Q$  from  $\mathcal{P}$ 
11    add  $Q$  to  $\mathcal{P}'$ 
12    compute coverage by  $\mathcal{P}'$  and append it to  $C$ 
13    compute shortest path from  $s(Q)$  to  $t(Q)$ , add this shortest path to  $\mathcal{P}_{SP}$ 
14    compute coverage by  $\mathcal{P}_{SP}$  and append it to  $C_{sp}$ 
15    perform random walk: start a random agent in  $s(Q)$ , agent moves randomly
        to one of the neighbors of current nodes (restricted by neighbor choice
        mechanism), until stopping criterion is reached.
16    add random walk to  $\mathcal{P}_{RAND}$ 
17    compute coverage by  $\mathcal{P}_{RAND}$  and append it to  $C_{rand}$ 
18  end
19  save  $C$ ,  $C_{sp}$  and  $C_{rand}$ 
20  reset  $\mathcal{P}$  to initially given set of walks
21 end

```

subset of trajectories (10% of all trajectories) was used instead of the complete set of trajectories. They do not show large differences. It is, however, interesting that shortest paths and random walks restricted by line changes, show faster coverage than the real trajectories and the random walks restricted by path length.

For all considered Rush Hour games (see Figure 3.10a), the set of shortest paths only covers a small subset of graph nodes. This is not surprising since all real trajectories start at the same node and end at one of a few goal nodes. The coverage by real trajectories and random walks is different for the three games: It is both higher than the coverage by shortest paths for all three games, but while their coverage is approximately the same for game C, the coverage by random walks is higher than by real trajectories for game A—and the other way around for game B.

Node usage We compared the node usage by real trajectories to the node usage of the generated random walks. Figures 3.11a to 3.11f show the node usage $nu(v)$ for each node v by the real trajectories and the random walks: The x -axis represents the network nodes, ordered by their node usage by real trajectories, while the y -axis shows the node's node usage by the corresponding walk type (shortest paths/random walks/real trajectories). Several observations can be made: For the Rush Hour games, there is only a low correlation between the node usage by real trajectories and by the random walks⁸ (Pearson correlation coefficient of 0.77 (game A), 0.64 (game B), and 0.36 (game

⁸random walks with uniform neighbor choice

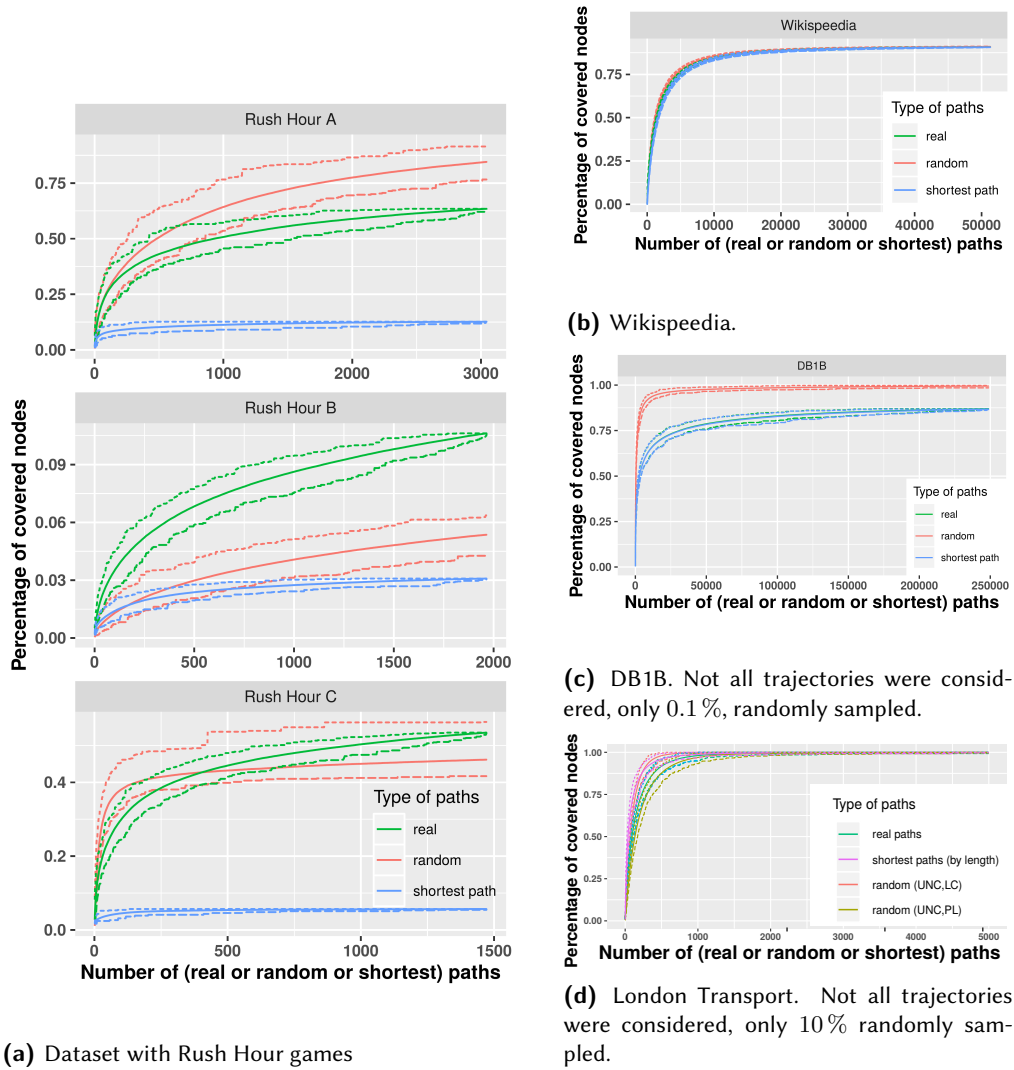


Figure 3.10 Coverage of the networks by real trajectories, shortest paths, and random walks. The solid line represents the mean coverage value over the $N = 500$ iterations, the dashed lines the minimal and maximal coverage values over the 500 iterations.

C)). For the datasets Wikipedia, DB1B, and London Transport, we found higher correlations, i.e., correlation coefficients between 0.81 and 0.87. This is also supported by Figure 3.11g where the node usage by the real trajectories is shown on the x -axis, and the node usage by the random walks with uniform neighbor choice and path length restriction, is shown on the y -axis. While for Wikipedia, London Transport, and DB1B, it can be seen that the node usages are correlated, this is not the case for the Rush Hour games. Note that for DB1B, London Transport, and Wikipedia, it is exactly the same set of nodes that are used most often by both types of walks, real trajectories and random walks (the appendix contains a figure showing the nodes with the highest node usage by real trajectories and random walks; see Figure A.1). By construction, random walks do not incorporate any knowledge about the network structure nor the target of each trajectory, while the real trajectories do not necessarily use the shortest path to their target, but the corresponding (real) agents clearly use their knowledge about the network structure in order to reach their goal quite fast. Nevertheless, the trajectories of both types of agents show a similar node usage pattern with respect to the question of which nodes are used most often. This implies that the node usage pattern, caused by real trajectories and by random walks, is mainly due to the established source

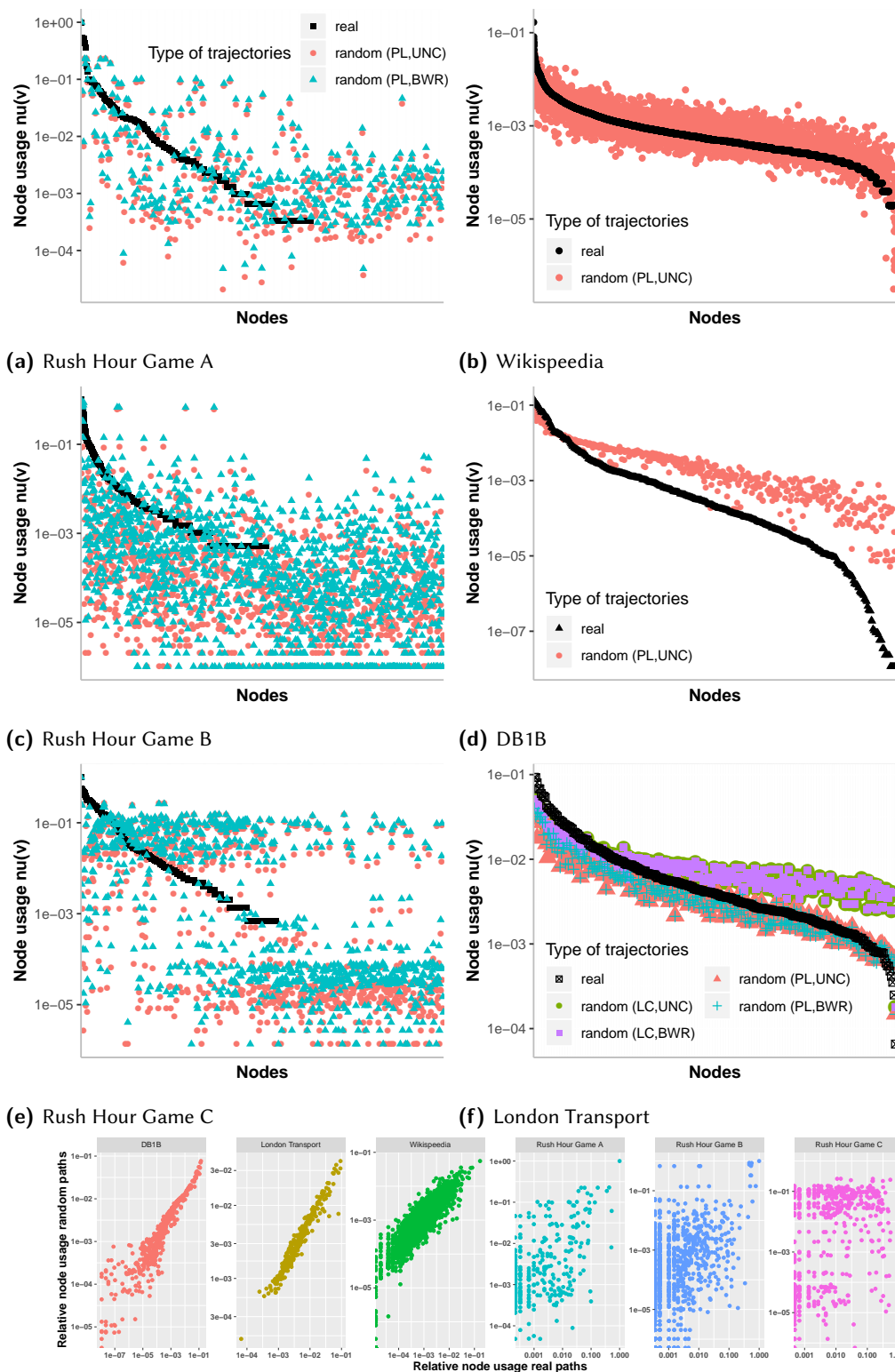
distribution because the source distribution was kept the same for the random walk generation (and probably mainly due to the source-target-distribution, but since only the source distribution can be kept in the random walk procedure, this cannot be tested).

Furthermore, it can be seen that for most datasets, the different variants of random walks do not yield distinct node usages: For the Rush Hour games, the node usage by random walks with backwards-restricted neighbor choice consistently yielded slightly higher node usages for each node than the random walks with uniform neighbor choice. For Wikispeedia and DB1B, the node usages by random walks with uniform neighbor choice and backwards-restricted neighbor choice yielded almost exactly the same values of node usage which is why the figure only shows the node usage of random walks with uniform neighbor choice. For London Transport, the variant of neighbor choice also yielded almost exactly the same values for node usage; however, the choice of the stopping criterion had an impact on the node usage values. Although the stopping criterion of line change restriction (LC) was introduced for this dataset in particular in order to gain more realistic random walks, it is actually the random walks with path length restriction for which the node usage pattern is more like the node usage pattern by real trajectories.

Although for the datasets London Transport, DB1B, and Wikispeedia, a high correlation of node usage by random walks and real trajectories can be observed, there is also another effect: The node usage distribution is "flattened" when considering random walks instead of real trajectories. While the nodes with the highest node usage by the real trajectories also have the highest node usage by the random walks, their node usage by the real trajectories is higher than their node usage by the random walks. At the other end of the scale, the nodes with the smallest node usage values for random walks and real trajectories were used even less often by the real trajectories than by the random walks. In other words, the range of node usage values is larger for the real trajectories than for the random walks: For the real trajectories, a few nodes were used very often and a few nodes were used very rarely, while for the random walks, these differences are less extreme. Figure 3.12 shows the cumulative node usage distributions for the real trajectories and for the random walks. For the Wikispeedia dataset, the cumulative node usage distributions are even coincident for most possible values; only for the smallest and largest possible values of the node usage are the cumulative distributions different. For DB1B and London Transport, the finding was the same as before: The nodes used least often show smaller node usage values for the real trajectories than for the random walks, and the nodes used most often show larger node usage by the real trajectories than by the random walks.

Interestingly, although the node usage by the random walks and by the real trajectories shows smaller correlations for the Rush Hour games, the cumulative distributions of the node usage are—except for the same "flattening effect" as in the remaining datasets—quite similar for the random walks and the real trajectories. This means that although the order of nodes by node usage is different for the random walks and the real trajectories, their cumulative distributions are alike.

Node pair usage We also considered the node pair usage of the node pairs by the random walks. Figure 3.13 shows the cumulative distribution of the node pair usage n_{pu} by the real trajectories (as shown in Section 3.3.1 in Figure 3.5c) and by the generated random walks, using uniform neighbor choice and path length restriction as stopping criteria. As the random walks were tied to the set of real trajectories \mathcal{P} , the source distribution was the same for the real trajectories and the random walks. However, Figure 3.13 shows that the source-target-distribution measured by node pair usage is different for the random walks and the real trajectories. As for node usage by the random walks, a "flattening" of the distribution can be observed here as well: The node pairs demanded most often by the real trajectories have higher node pair usage than the node pairs demanded most often by the random walks, and the node pairs demanded least often by the real trajectories have lower node pair usage than the node pairs demanded least often by the random walks.



(g) Node usage of the real paths vs mean node usage of the random walks (UNC, PL). Node usage is normalized by the number of (real or random) agents.

Figure 3.11 Node usage by real trajectories and by random walks: For panels (a)-(f), each point represents one node and its node usage (by random walks or real trajectories), ordered by their node usage by real trajectories. For DB1B and Wikipedia, only results of random walks with uniform neighbor choice are shown since the results for backwards-restricted neighbor choice are qualitatively the same. Panel (g) shows the node usage of each node by the real trajectories (on the x -axis) versus its node usage by the random walks (y -axis).

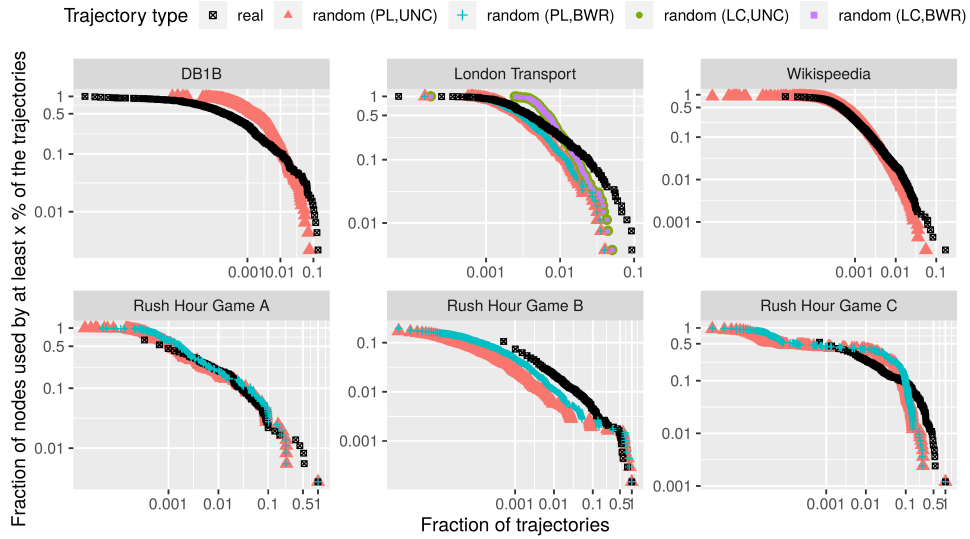


Figure 3.12 Cumulative node usage distribution of real trajectories and random walks. The random walks implement a uniform neighbor choice (UNC) or a backwards-restricted neighbor choice (BWR), their length is restricted by the length (PL) or, if applicable, by the weight (PW) of the corresponding real trajectory. Note the logarithmic scales on both axes.

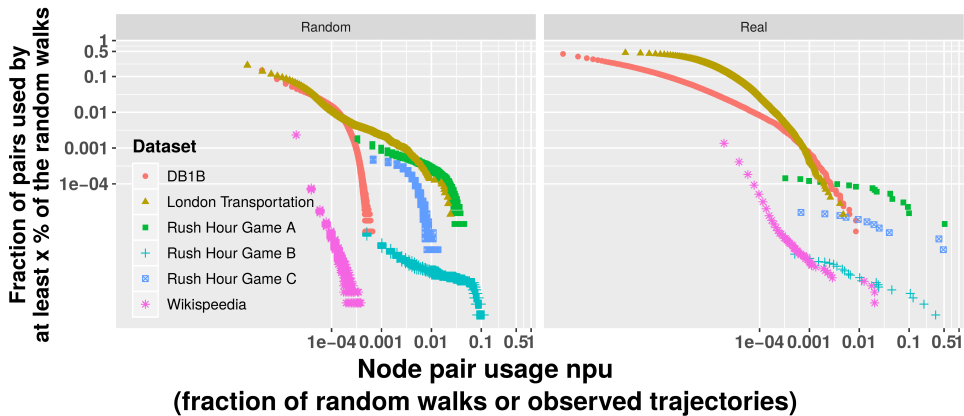


Figure 3.13 Cumulative distribution of source-target-frequency of the random walks (left, with uniform neighbor choice (UNC) and path length restriction (PL) as stopping criteria) and of the real trajectories (right).

3.6 Summary

In this chapter, datasets containing real-world network flows, i.e., two datasets of passenger flows in transportation systems, one dataset containing human navigation in information networks, and one dataset containing game logs for a board game, were studied. Several network measures, particularly classic centrality measures, implicitly contain a model of a network flow with certain properties: Centrality measures then indicate the importance of a node with respect to network flows with the assumed properties. Therefore, in this chapter, we investigated to which extent real-world network flows match the assumed properties. We found that all considered datasets deviated considerably from most assumed properties. Although for all considered network flows, the assumption of shortest paths seems to be given according to the process description, the data revealed that there is a considerable number of trajectories that are longer than the respective shortest path. Furthermore, we found a large imbalance in the distribution of network flow between the node pairs: For all datasets, most of the flow is accumulated between only a few node pairs, while there is no or only a small amount of flow between most node pairs. This is substantially different than assumed by standard centrality measures, which assign equal importance to each node pair. Since the real-world trajectories showed different properties than the corresponding shortest paths, in a second step, we examined whether their properties can be reproduced by a simple process model, i.e., a random-walk-based model. For this purpose, we implemented several variants of random walk simulations that maintained certain properties of the real-world network flows, but moved randomly through the network. We found that the real-world network flows resembled the random walks in some aspects, but were different in others.

In order to investigate how deviations from assumptions impact the results of centrality measures, the next chapter will introduce flow-based centrality measures that incorporate the properties of real-world network flows into the measures.

Flow-based centrality measures

Chapter outline

In the previous chapter, the analysis of real-world datasets containing network flows revealed an extreme imbalance in the usage of the network by the flow: It was seen that not all nodes are visited by the flow, and that a few nodes are used very often, while most nodes are used only a few times. The same holds for the node pairs: A few node pairs were found to be the source and target of many trajectories while most node pairs were used only rarely or not at all as source and target. Furthermore, *how* the observed trajectories move through the network was neither via shortest paths nor via random walks. These observations have an impact on common centrality measures for networks. Classic centrality measures assume the presence of network flow and contain implicit assumptions about the properties of the network flow. It can be expected that centrality measures quantifying the nodes' importance with respect to a hypothetical network flow with "ideal" properties will identify different nodes as the most central ones than if the nodes' importance is measured with respect to the real network flow properties. In this chapter, we will analyze the impact on the nodes' centrality rankings when the properties of the real-world flow are used instead of the hypothetical properties of the network flow. For this reason, we will introduce several *flow-based* closeness and betweenness variants that either use the hypothetical properties of the network flow (such as equal amount of flow between each node pair or usage of shortest paths) or use the observed properties of the real-world network flow. By introducing several variants that can "switch" between the assumed and the real properties of the network flow, we are able to analyze in detail which assumptions of the centrality measures have an impact on the results.

The results presented in this chapter are mainly based on [4] and [6].

4.1 Motivation

When analyzing a complex system as a network, it is often of interest to identify the most important actor in the system. When modeling the system as a network, there exist centrality measures that were designed to compute a score of importance for every node according to its position in the network. Thus, a centrality measure is a function assigning a number to each node of the network, using solely the structure of the network. From the few named examples, it is clear that there are different requirements for methods identifying important actors which is why there exist dozens of different centrality measures. The best-known ones are degree centrality [Nie74], closeness centrality [Fre78], betweenness centrality [Fre77, Ant71], Eigenvector centrality [Bon72], and Katz centrality [Kat53]. Faced with such a wealth of measures, the question arises which of these measures is applicable in which situation. In order to answer this question, it is helpful to consider how these measures are actually computed. Although all these measures only use the structure of the network, they all assume a process flowing through the network. This was already stated

by Freeman in his article presenting betweenness-based measures [Fre77]: “Thus, the use of these three measures is appropriate only in networks where betweenness may be viewed as important in its potential for impact on the process being examined.” Thus, centrality measures quantify the importance of a node with respect to a process, i.e., entities flowing through the network using the network structure. As described in Chapter 2, there are two major points regarding the usability of centrality measures, pointed out by Borgatti [Bor05]: (i) The most popular centrality measures assume the *presence* of a process, and (ii) they also assume certain properties of such processes. In other words, each centrality measure implicitly incorporates the model of a flow process with certain properties. He found that most well-known centrality measures assume that the network processes move on shortest paths or are transmitted by parallel duplication (see also Chapter 2). It is obvious, however, that most processes for which the most important actors are of interest do not have these properties; for example information flow or infection spreading. Even worse, even those network flows for which a transfer process and usage of shortest paths can be assumed, do not exhibit this behavior in reality. In Chapter 3, several datasets of network flows with a transfer process and presumably shortest paths were analyzed in detail. Although the entities tracked in the datasets have a target that they try to reach as fast as possible, we found that even these network flows violate the assumption of shortest paths.

Which consequences does this have for centrality measures? Consider Figure 4.1 as an example: It shows a small network with two example network flows sketched with blue and orange edges. Standard betweenness and closeness centrality would assign high values to the nodes A and B since, for betweenness centrality, (almost) all shortest paths between the two groups involve A and B, and, for closeness centrality, these nodes are closest to nodes of *both* groups. Let us assume a network flow flowing along the bold blue edges. Although for this flow, it is true that there is flow between the two groups, it does not use the shortest path from its origin to its destination. The nodes A and B, which are ranked high by standard measures, are not relevant at all with respect to the network flow at hand. Another case occurs for the flow sketched with dashed orange edges. This flow moves along shortest paths; however, it is only relevant for the right subgroup of nodes. For this toy network flow, node F is highly relevant, which cannot be recognized by the standard measures because they assume a simple process model in which an equal quantity of flow is assumed between each pair of nodes.

If even network flows for which the classic centrality measures are applicable in the classification of Borgatti [Bor05] do not show these properties in reality, the question arises how valid the results of any centrality measure can be. If a centrality measure is applied on a network assuming a network process with certain properties, but the actual network flow does not show these assumed properties, the centrality measure cannot be expected to be able to identify the most important actor with respect to the actual network process. Or can it?

In this chapter, the goal is to answer exactly this question:

Which impact on the resulting node rankings does it have if an empiric network flow does not have the characteristics of the process model incorporated in the centrality measure?

In order to answer this question, our approach is as follows: Instead of incorporating a process *model* into a centrality measure, we incorporated the characteristics of the *actual* empirically observed network flow. For example, instead of counting in how many shortest paths a node is contained, we counted in how many actually observed trajectories a node is contained.

We can do this even more systematically and test every single assumption contained in a centrality measure on its own: When looking at the formula of betweenness centrality, it can be seen that besides the usage of shortest paths, there are more assumptions in the process model: There is a

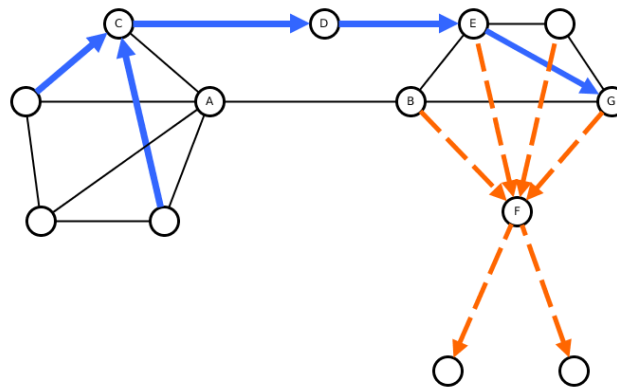


Figure 4.1 An example network with a simple network flow illustrated by colored edges. Standard betweenness centrality would assign a high value to nodes A and B because their position is between the two larger subgraphs and all shortest paths between the subgraphs contain the nodes A and B. If, however, an example network flow such as the one sketched with bold blue edges, does not use these edges, standard betweenness centrality is not able to identify the most important node with respect to the existing network flow.

flow between every pair of nodes, and the flow is of equal frequency for every node pair. In the datasets with empirically observed flow data, we found that these assumptions were not generally true: There were node pairs between which no flow was observed, and there was a large amount of flow between a few node pairs, while there was only a small amount of flow between most node pairs.

In order to answer the main question, we introduce different variants of centrality measures, either using the property of the incorporated process model or using the actual characteristics of the empirically observed network flow. This allows us to systematically “switch on or off” specific assumptions in the process model by replacing them with the actual flow behavior.

The structure of this chapter is thus as follows: Based on the centrality measures’ assumptions reviewed in Chapter 2, Section 4.2 introduces variants of betweenness, and closeness centrality that are able to include or exclude empirically observed flow behavior instead of the process model. In Section 4.3, we briefly review existing methods for comparing values and rankings of centrality or other scoring measures, and introduce a new measure for comparing two rankings: the weighted overlap coefficient. In Section 4.4, we describe the datasets used, before presenting our findings in the remaining sections. The results sections are structured along the following questions:

- (i) How robust are the standard measure variants, i.e., standard betweenness, and standard closeness centrality against variations in their process model? In Section 4.5, we investigate to which extent the rankings of the flow-based centrality variants differ from the ranking of the corresponding standard centrality measure.
- (ii) Which nodes are impacted by changes in the process model? This question is analyzed in Section 4.6 in two steps: We first consider those nodes that are ranked highly by at least one measure variant because most of the time, one is interested in the *most important* entities. Changes in ranking among the high-ranked nodes are of higher relevance than changes in ranking among less highly-ranked nodes. In a second step, we focus on those nodes that are impacted most by including or excluding a certain aspect of the empirically observed flow. If possible, we provide explanations on why these nodes experience such a large change in ranking position.
- (iii) Which assumptions matter and which do not? By comparing the rankings of the flow-based centrality variants including or excluding certain assumptions, we can analyze whether there are some assumptions whose violations have a large impact on the resulting rankings or

whether their impact is minor. In this context, we need to consider to which extent the assumptions are actually violated. Section 4.8 presents our results for this question.

Notation In general, given a graph $G = (V, E)$, a centrality measure is a function $c : V \rightarrow \mathbb{R}$ that assigns a value to each node. Usually, a centrality measure is required to be a structural index, i.e., its values solely depend on the structure of the graph. Normally, a high value of c indicates high importance of the node while a low value indicates low importance. The values of c induce a ranking on the nodes where each node is assigned a rank from 1 to $|V|$: The node with the highest c value gets the rank 1, etc. A node with a high c value is “ranked high” while a node with a small c value is “ranked low”. In this chapter, we will consider closeness centrality as defined in Definition 2.8 on page 14, and betweenness centrality including endpoints as defined in Definition 2.10 on page 15.

4.2 Flow-based centrality measures

In Chapter 3, we have shown that the main assumptions of the considered centrality measures (as described in Chapter 2) are not met in empirical flow data: Flows do not move along shortest paths through the network, and there are node pairs in the network between which there is more communication than between others.

The question arises whether this actually matters for the identification of the most important nodes by centrality measures. Since the network structure and the corresponding network flow are often not independent of each other—actual demand by the flow process might shape the network and the flow process is constrained by the network structure—, it is possible that the network structure is indeed sufficient. On the other hand, the contained simple process model might be too simplistic to capture those nodes that are relevant for the actual process.

Since datasets containing network flows are available, the information of how a process actually flows through the network is accessible. We thus used the following approach: For each centrality measure, the following sections will introduce *flow-based* variants that—instead of the simple process model—incorporate the behavior of the real-world network flow. For example, one variant will count in how many empirically observed trajectories a node is contained instead of counting the number of shortest paths it is contained in. With this approach, we are able to “switch on and off” the existing assumptions contained in the centrality indices and replace the behavior of the simple process model with the behavior of the real-world process flow.

4.2.1 Flow-based betweenness centrality

We will introduce four flow-based betweenness variants which are all based on the empirical flow data to a certain extent. For all variants, the empirical flow data is needed which is assumed to be given as a set of trajectories in the graph, $\mathcal{P} = \{P_1, \dots, P_\ell\}$ where elements of \mathcal{P} are walks in the graph G .

As a general framework, we introduce a weighted betweenness centrality:

Definition 4.1: Generalized weighted betweenness centrality

The weighted betweenness centrality is defined as

$$B_w(v) = \sum_{s \in V} \sum_{t \in V} w(s, t, v) \cdot \frac{\sigma_{st}(v)}{\sigma_{st}}$$

with a weight function $w : V \times V \times V \rightarrow \mathbb{R}$.

Standard betweenness centrality is obtained by inserting the weight function

$$w(s, t, v) = \begin{cases} 0 & \text{if } s = t \text{ or } s = v \text{ or } v = t \\ 1 & \text{else} \end{cases}$$

and standard betweenness centrality including endpoints has the weight function $w_E(s, t, v) = 1$ for all $s, t, v \in V$.

Figure 4.2 shows an example graph with six paths in it, indicated by directed edges with different colors. Hence, the set of paths is here given as

$$\mathcal{P} = \{P_1, P_2, P_3, P_4, P_5, P_6\}$$

with

$$\begin{aligned} P_1 &= (10, 1, 2, 3, 4, 6), \\ P_2 &= (7, 3, 4, 6), \\ P_3 &= (8, 7, 2, 3, 9, 4, 6), \\ P_4 &= (7, 3, 9, 4, 6), \\ P_5 &= (1, 2, 3, 9, 4), \text{ and} \\ P_6 &= (7, 2, 3, 9, 4, 6). \end{aligned}$$

Color and size of the nodes in the figure indicate the node's centrality value with respect to the corresponding measure, while the numbers next to the nodes show the corresponding measure value. Figure 4.2b show the betweenness centrality values of the nodes with respect to the standard betweenness centrality that includes the endpoints of the shortest paths.

Table 4.1 shows an overview of the introduced flow-based betweenness measures.

Note that the subsequent analysis will be mainly performed on the rankings induced by the measure variants and not on the actual values which is why there is no need for normalization of the measures.

Betweenness variant B_S The first variant will keep the assumption of shortest paths (indicated by the subscript S), but will only use the (shortest) paths between those node pairs between which real-world flow has been observed. Consider again Figure 4.1 as a motivating example: The process flow indicated by the orange edges only takes place within the right group of nodes which is why there is no reason to consider nodes A or B as "gatekeepers" for the present flow. Whether this aspect is relevant for real-world network flows, is tested with the variant B_S , where we count only shortest paths between node pairs that are source and target of the real-world network flow.

Definition 4.2: Betweenness variant B_S

Thus, we define

$$B_S(v) = \sum_{s \in V} \sum_{t \in V} w_S(s, t, v) \cdot \frac{\sigma_{st}(v)}{\sigma_{st}} \quad (4.1)$$

with the weight function

$$w_S(s, t, v) = \begin{cases} 1 & \text{if } \exists P \in \mathcal{P} : s(P) = s \text{ and } t(P) = t \\ 0 & \text{else} \end{cases}$$

In the example in Figure 4.2c, the weights are hence $w_S(1,4,\cdot) = w_S(7,6,\cdot) = w_S(8,6,\cdot) = w_S(10,6,\cdot) = 1$ and $w_S(s,t,v) = 0$ for all other $s,t,v \in V$. For node 3, the centrality value is then

$$\begin{aligned}
 B_S(3) &= \underbrace{w_S(1,4,3)}_1 \cdot \underbrace{\frac{\sigma_{1,4}(3)}{\sigma_{1,4}}}_{=\frac{1}{1}} + \underbrace{w_S(8,6,3)}_{=1} \cdot \underbrace{\frac{\sigma_{8,6}(3)}{\sigma_{8,6}}}_{=\frac{0}{1}} + \\
 &\quad \underbrace{w_S(10,6,3)}_{=1} \cdot \underbrace{\frac{\sigma_{10,6}(3)}{\sigma_{10,6}}}_{=\frac{1}{1}} + \underbrace{w_S(7,6,3)}_{=1} \cdot \underbrace{\frac{\sigma_{7,6}(3)}{\sigma_{7,6}}}_{=\frac{1}{2}} \\
 &= 2.5
 \end{aligned}$$

Table 4.1 Categorization of the introduced flow-based betweenness centralities

	count	how?	sum over	weight
B_S	shortest		$s, t: \exists P \in \mathcal{P} : s(P) = s \rightarrow t = t(P)$	1
B_{SW}	shortest		$s, t: \exists P \in \mathcal{P} : s(P) = s \rightarrow t = t(P)$	$\#P : s \rightarrow t$
B_R	real	$\rightarrow s \rightarrow v \rightarrow t \rightarrow$	all nodes	1
B_{RW}	real	$s \rightarrow v \rightarrow t$	$s, t: \exists P \in \mathcal{P} : s(P) = s \rightarrow t = t(P)$	$\#P : s \rightarrow t$

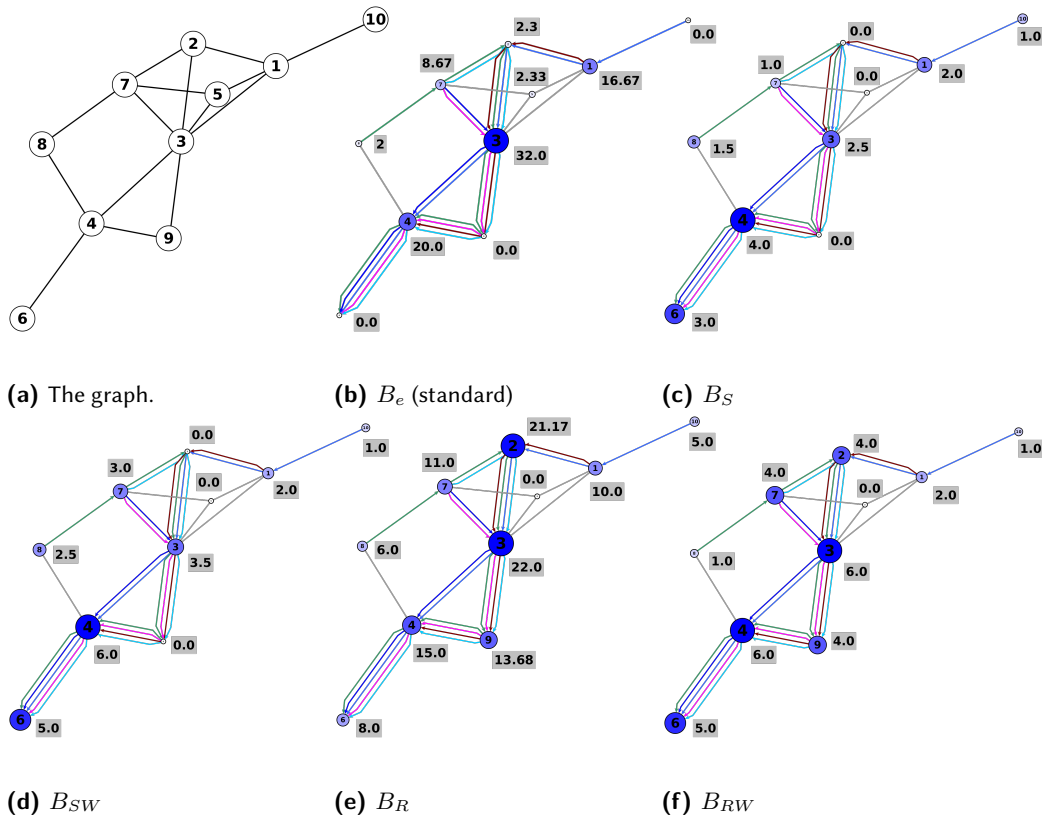


Figure 4.2 Application of the flow-based betweenness measures on an example graph with the set $\mathcal{P} = \{P_1, P_2, P_3, P_4, P_5, P_6\}$ with $P_1 = (10, 1, 2, 3, 4, 6)$, $P_2 = (7, 3, 4, 6)$, $P_3 = (8, 7, 2, 3, 9, 4, 6)$, $P_4 = (7, 3, 9, 4, 6)$, $P_5 = (1, 2, 3, 9, 4)$, and $P_6 = (7, 2, 3, 9, 4, 6)$. Size and colour of the nodes correspond to their centrality value, the values of the centrality measures are shown in the grey box next to each node.

Betweenness variant B_{SW} In this variant, the assumption of shortest paths is still kept, but the assumption of equal amount of flow between each node pair is dropped. We have seen that between some node pairs, there is larger flow than between others. The idea is that a node should get a higher centrality value if it is on many shortest paths between node pairs that are in high demand as source-target-pairs by the real-world flow than if a node is on many shortest paths between node pairs where demand is less. In the previous variant, however, any node pair can contribute at most the value 1 to the centrality of a node, regardless how often (if at least once) this pair was demanded as a source and destination of the process. Therefore, a weight (thus the subscript W) is chosen such that being between highly demanded node pairs yields a higher centrality score. The weight is thus proportional to the number of times this node pair was the source and destination of an empirically observed trajectory.

Definition 4.3: Betweenness variant B_{SW}

We define

$$B_{SW}(v) = \sum_{s \in V} \sum_{t \in V} w_{SW}(s, t, v) \cdot \frac{\sigma_{st}(v)}{\sigma_{st}} \quad (4.2)$$

with the weight function^a

$$w_{SW}(s, t, v) = |\{P \in \mathcal{P} | s(P) = s, t(P) = t\}|$$

^aSince we are not interested in the actual values of the centrality measures, but in the node rankings they yield, it is not necessary to require the weights to be in the interval $[0, 1]$ and therefore, we do not need to normalize w_{SW} .

For the example in Figure 4.2d, this means that the weights are as follows

$$w_{SW}(s, t, v) = \begin{cases} 1 & \text{if } (s=1, t=4) \text{ or } (s=8, t=6) \text{ or } (s=10, t=6) \\ 3 & \text{if } s=7, t=6 \\ 0 & \text{otherwise} \end{cases}$$

which then yields for node 3

$$\begin{aligned} B_{SW}(3) &= \underbrace{w_{SW}(1, 4, 3)}_{=1} \cdot \underbrace{\frac{\sigma_{1,4}(3)}{\sigma_{1,4}}}_{=\frac{1}{1}} + \underbrace{w_{SW}(8, 6, 3)}_{=1} \cdot \underbrace{\frac{\sigma_{8,6}(3)}{\sigma_{8,6}}}_{=\frac{0}{1}} + \\ &\quad \underbrace{w_{SW}(10, 6, 3)}_{=1} \cdot \underbrace{\frac{\sigma_{10,6}(3)}{\sigma_{10,6}}}_{=\frac{1}{1}} + \underbrace{w_{SW}(7, 6, 3)}_{=3} \cdot \underbrace{\frac{\sigma_{7,6}(3)}{\sigma_{7,6}}}_{=\frac{1}{2}} \\ &= 3.5 \end{aligned}$$

Since shortest paths are counted, frequently visited nodes such as node 9 or 2, which are not on any shortest path, get a low centrality value.

Betweenness variant B_R Unlike the previous two measure variants which count in how many *shortest* paths a node is contained, the following two measures count in how many *observed* trajectories a node is contained (indicated by the subscript R for *real* trajectories). Consider the example network flow in Figure 4.1 indicated by the blue edges. Instead of using the shortest path via nodes A and B, it uses the detour via nodes C, D, and E which is why those nodes should be rated as relevant for the present process. This cannot be accomplished by any of the previous measure variants counting shortest paths. For this reason, we need to define a flow-based version of σ_{st} and

$\sigma_{st}(v)$ as the number of observed trajectories between two nodes and that contain v . In order to keep the assumption that there is communication between any pair of nodes and not only between the node pairs that are actually source and destination of real trajectories, we define $\sigma_{st}^{\mathcal{P}}$ as the number of observed trajectories that *contain* s and t (in this order) and $\sigma_{st}^{\mathcal{P}}(v)$ as the number of observed trajectories that contain s and t and v in between. Otherwise, if $\sigma_{st}^{\mathcal{P}}$ was defined as the number of observed trajectories *from* s *to* t , node pairs that are not source and target of any observed trajectory would not be considered at all, meaning the assumption of equal amount of flow between each node pair would already be dropped. It is clear that even with the given definition of $\sigma_{st}^{\mathcal{P}}$ and $\sigma_{st}^{\mathcal{P}}(v)$, node pairs (s, t) where s or t are not contained in any observed trajectory (or not contained in at least one common observed trajectory), will not contribute to the centrality value because $\sigma_{st}^{\mathcal{P}}$ is 0 in this case.

Definition 4.4: Betweenness variant B_R

We define

$$B_R(v) = \sum_{s \in V} \sum_{t \in V} w_R(s, t, v) \cdot \frac{\sigma_{st}^{\mathcal{P}}(v)}{\sigma_{st}^{\mathcal{P}}} \quad (4.3)$$

with the weight function $w_R(s, t, v) = 1$ for all $s, t, v \in V$ with $s \neq t$, the convention $\frac{0}{0} = 0$, and

$$\sigma_{st}^{\mathcal{P}} = |\{P \in \mathcal{P} | P = (\dots, s, \dots, t, \dots)\}|$$

and

$$\sigma_{st}^{\mathcal{P}}(v) = |\{P \in \mathcal{P} | P = (\dots, s, \dots, v, \dots, t, \dots)\}|$$

In the example in Figure 4.2e, all nodes except node 5 are contained in some path. For node 2, we therefore get

$$\begin{aligned} B_R(2) &= \underbrace{\frac{\sigma_{1,4}^{\mathcal{P}}(2)}{\sigma_{1,4}^{\mathcal{P}}}}_{=\frac{2}{2}} + \underbrace{\frac{\sigma_{2,6}^{\mathcal{P}}(2)}{\sigma_{2,6}^{\mathcal{P}}}}_{=\frac{3}{3}} + \underbrace{\frac{\sigma_{1,9}^{\mathcal{P}}(2)}{\sigma_{1,9}^{\mathcal{P}}}}_{=\frac{1}{1}} + \underbrace{\frac{\sigma_{2,3}^{\mathcal{P}}(2)}{\sigma_{2,3}^{\mathcal{P}}}}_{=\frac{4}{4}} + \underbrace{\frac{\sigma_{8,2}^{\mathcal{P}}(2)}{\sigma_{8,2}^{\mathcal{P}}}}_{=\frac{1}{1}} + \underbrace{\frac{\sigma_{10,4}^{\mathcal{P}}(2)}{\sigma_{10,4}^{\mathcal{P}}}}_{=\frac{1}{1}} \\ &+ \underbrace{\frac{\sigma_{7,3}^{\mathcal{P}}(2)}{\sigma_{7,3}^{\mathcal{P}}}}_{=\frac{2}{4}} + \underbrace{\frac{\sigma_{8,9}^{\mathcal{P}}(2)}{\sigma_{8,9}^{\mathcal{P}}}}_{=\frac{1}{1}} + \underbrace{\frac{\sigma_{2,4}^{\mathcal{P}}(2)}{\sigma_{2,4}^{\mathcal{P}}}}_{=\frac{4}{4}} + \underbrace{\frac{\sigma_{10,3}^{\mathcal{P}}(2)}{\sigma_{10,3}^{\mathcal{P}}}}_{=\frac{1}{1}} + \underbrace{\frac{\sigma_{7,4}^{\mathcal{P}}(2)}{\sigma_{7,4}^{\mathcal{P}}}}_{=\frac{2}{4}} + \underbrace{\frac{\sigma_{1,2}^{\mathcal{P}}(2)}{\sigma_{1,2}^{\mathcal{P}}}}_{=\frac{2}{2}} \\ &+ \underbrace{\frac{\sigma_{10,6}^{\mathcal{P}}(2)}{\sigma_{10,6}^{\mathcal{P}}}}_{=\frac{1}{1}} + \underbrace{\frac{\sigma_{1,3}^{\mathcal{P}}(2)}{\sigma_{1,3}^{\mathcal{P}}}}_{=\frac{2}{2}} + \underbrace{\frac{\sigma_{7,6}^{\mathcal{P}}(2)}{\sigma_{7,6}^{\mathcal{P}}}}_{=\frac{2}{4}} + \underbrace{\frac{\sigma_{8,6}^{\mathcal{P}}(2)}{\sigma_{8,6}^{\mathcal{P}}}}_{=\frac{1}{1}} + \underbrace{\frac{\sigma_{8,3}^{\mathcal{P}}(2)}{\sigma_{8,3}^{\mathcal{P}}}}_{=\frac{1}{1}} + \underbrace{\frac{\sigma_{2,9}^{\mathcal{P}}(2)}{\sigma_{2,9}^{\mathcal{P}}}}_{=\frac{3}{3}} \\ &+ \underbrace{\frac{\sigma_{10,2}^{\mathcal{P}}(2)}{\sigma_{10,2}^{\mathcal{P}}}}_{=\frac{1}{1}} + \underbrace{\frac{\sigma_{8,4}^{\mathcal{P}}(2)}{\sigma_{8,4}^{\mathcal{P}}}}_{=\frac{1}{1}} + \underbrace{\frac{\sigma_{7,2}^{\mathcal{P}}(2)}{\sigma_{7,2}^{\mathcal{P}}}}_{=\frac{2}{2}} + \underbrace{\frac{\sigma_{7,9}^{\mathcal{P}}(2)}{\sigma_{7,9}^{\mathcal{P}}}}_{=\frac{2}{2}} + \underbrace{\frac{\sigma_{1,6}^{\mathcal{P}}(2)}{\sigma_{1,6}^{\mathcal{P}}}}_{=\frac{1}{1}} = 21.17 \end{aligned}$$

We see that nodes whose centrality value was very small before because they are not on (many) shortest paths have a (relatively) larger centrality value in this variant. For example, node 9 or node 2 have minor importance with respect to the standard betweenness centrality, but rise in importance in this variant because they are used by a certain number of real paths.

Betweenness variant B_{RW} The last betweenness measure variant combines the idea of making use of a weight function that is proportional to the number of observed trajectories using this s - t -pair as source and destination, and the idea of counting the observed trajectories instead of the shortest paths in which a node v is contained. Therefore, both assumptions, usage of shortest paths and equal amount of flow between each node pair, are dropped and replaced by the behavior of the empirical flow. Since in this case, the weight function will yield 0 for node pairs (s, t) where s and t are not source and target of any observed trajectory, we can use a simpler flow-based version of σ_{st} and $\sigma_{st}(v)$.

Definition 4.5: Betweenness variant B_{RW}

We define

$$B_{RW}(v) = \sum_{s \in V} \sum_{t \in V} w_{RW}(s, t, v) \cdot \frac{\sigma_{st}^{\mathcal{P}}(v)}{\sigma_{st}^{\mathcal{P}}} \quad (4.4)$$

with the weight function $w_{RW}(s, t, v) = w_{SW}(s, t, v)$. and $\sigma_{st}^{\mathcal{P}} = |\{P \in \mathcal{P} | s(P) = s, t(P) = t\}|$ and $\sigma_{st}^{\mathcal{P}}(v) = |\{P \in \mathcal{P} | s(P) = s, t(P) = t, v \in P\}|$. This can be simplified to

$$\begin{aligned} B_{RW}(v) &= \sum_{s \in V} \sum_{t \in V} |\{P \in \mathcal{P} | s(P) = s, t(P) = t\}| \cdot \frac{\sigma_{st}^{\mathcal{P}}(v)}{\sigma_{st}^{\mathcal{P}}} \\ &= \sum_{s \in V} \sum_{t \in V} |\{P \in \mathcal{P} | s(P) = s, t(P) = t\}| \cdot \frac{|\{P \in \mathcal{P} | s(P) = s, t(P) = t, v \in P\}|}{|\{P \in \mathcal{P} | s(P) = s, t(P) = t\}|} \\ &= \sum_{s \in V} \sum_{t \in V} |\{P \in \mathcal{P} | s(P) = s, t(P) = t, v \in P\}| \\ &= |\{P \in \mathcal{P} | v \in P\}| \end{aligned}$$

which is the number of paths in \mathcal{P} that contain v .

B_{RW} is thus a kind of stress betweenness centrality.

We see in Figure 4.2f that node 3, which is central with respect to all previously considered centrality measures is also central with respect to B_{RW} , but nodes such as 9 and 2 rise in importance because they were actually used by a considerable number of trajectories. Figure 4.2 also shows that the different flow-based variants we introduced do change the relative importance of nodes in the given example. Sections 4.5 to 4.8 will present the results after the computation of the introduced measures using real-world empirical process flows.

4.2.2 Flow-based closeness centrality

For closeness centrality, there are some similar assumptions as for betweenness centrality which can be replaced by the behavior of the real-world process flow in flow-based centrality variants: usage of shortest paths and equal amount of flow from/to any other node. Similar as for the flow-based betweenness variants, we will introduce flow-based closeness variants that drop or keep these assumptions separately (as far as possible). We again begin with a generalized closeness centrality from which the different variants can then be derived¹.

¹In the following, for the sake of better readability, we will only introduce variants as in-closeness; the corresponding out-closeness can be easily derived.

Definition 4.6: Generalized weighted closeness centrality

Let

$$C_\omega(v) = \sum_{w \in V(v)} \frac{N(v)}{\omega(w, v) \delta(w, v)} \quad (4.5)$$

a generalized weighted closeness centrality with a weight function $\omega : V \times V \rightarrow \mathbb{R}$, a normalization factor $N : V \rightarrow \mathbb{N}$ and a distance function $\delta : V \times V \rightarrow \mathbb{R}$ and v -based subset of nodes.

Setting $N(v) := |V| - 1$ for all $v \in V$, $\delta(v, w) = d(v, w)$ as the graph-theoretic distance of nodes, $\omega(v, w) = 1$ for all $v, w \in V$, and $V(v) := V \setminus \{v\}$ yields the standard closeness (as defined above).

The following sections will introduce six different flow-based closeness variants. Their naming follows the same scheme as for the betweenness variants: The subscript S or R indicates whether shortest or real trajectories are considered, the additional subscript W indicates whether the amount of flow between two nodes is considered by the weight function. Table 4.2 gives an overview of the introduced closeness variants, Figure 4.3 shows a graph with example paths in it (the same as in the example for the betweenness variants), and the results of the flow-based closeness variants.

Closeness variant C_S The first flow-based closeness variant keeps the assumption of shortest paths (which is why the "normal" graph-theoretic distance function d can be used as a distance function), but only distances between those node pairs are considered between which a real-world flow is observed. There are several possibilities for defining which node pairs (s, t) to be considered: Either only node pairs (s, t) for which there exists a real trajectory starting in s and ending in t , or node pairs (s, t) such that at least one real trajectory starts in s and at least one real trajectory ends in t , or node pairs (s, t) such that s and t are contained in at least one common trajectory (in this order). We decided to introduce two variants, one using the first and one using the last definition (which will be the variant C'_S , defined in the following paragraph). In the variant to be defined here, *flow from s to t* is defined strictly as the existence of at least one observed trajectory starting in s and ending in t .

Definition 4.7: Closeness variant C_S

We define

$$C_S^*(v) = \sum_{w \in V_S(v)} \frac{N_S(v)}{\omega_S(w, v) d(w, v)} \quad (4.6)$$

with the weight function

$$\omega_S(w, v) = \begin{cases} 1 & \text{if } \exists P \in \mathcal{P} : s(P) = w, t(P) = v \\ 0 & \text{otherwise} \end{cases}$$

and the normalization factor

$$N_S(v) = |\{w \in V \mid \exists P \in \mathcal{P} : s(P) = w, t(P) = v\}|,$$

and $V_S(v) = \{w \in V \mid \omega_S(w, v) \neq 0\}$.

It is clear that only nodes in which at least one trajectory from \mathcal{P} ends, have a value above 0. In the example in Figure 4.3, all example paths end in node 6 or 4 which is why the values for all other

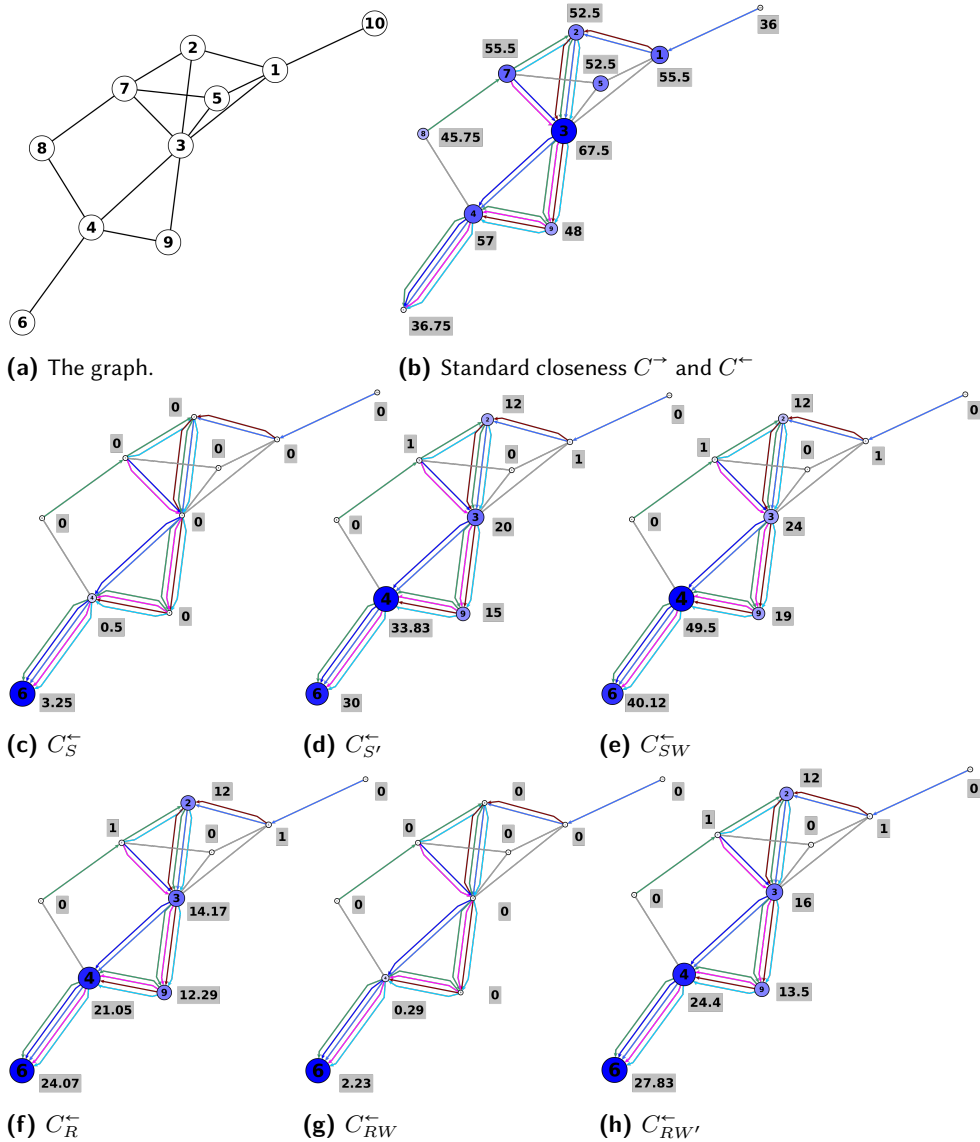


Figure 4.3 Application of the flow-based closeness (in-)measures on an example graph with the set $\mathcal{P} = \{P_1, P_2, P_3, P_4, P_5, P_6\}$ with $P_1 = (10, 1, 2, 3, 4, 6)$, $P_2 = (7, 3, 4, 6)$, $P_3 = (8, 7, 2, 3, 9, 4, 6)$, $P_4 = (7, 3, 9, 4, 6)$, $P_5 = (1, 2, 3, 9, 4)$, and $P_6 = (7, 2, 3, 9, 4, 6)$. Size and color of the nodes correspond to their centrality value, the values of the centrality measures are shown in the grey box next to each node. Only out-variants are shown here.

Table 4.2 Categorization of the introduced flow-based (in-)closeness centralities.

	Which distances?	To which nodes?	Weight
C_S	shortest	$\exists P : s(P) = w \rightarrow v = t(P)$	1
$C_{S'}$	shortest	$\exists P : \rightarrow w \rightarrow v \rightarrow$	1
C_{SW}	shortest	$\exists P : \rightarrow w \rightarrow v \rightarrow$	$\#P : \rightarrow w \rightarrow v \rightarrow$
C_R	real	$\exists P : \rightarrow w \rightarrow v \rightarrow$	1
C_{RW}	real	$\exists P : s(P) = w \rightarrow v = t(P)$	$\#P : w \rightarrow v$
$C_{RW'}$	real	$\exists P : \rightarrow w \rightarrow v \rightarrow$	$\#P : \rightarrow w \rightarrow v \rightarrow$

nodes are 0. For node 6, we obtain $V_S(6) = \{10, 7, 8\}$ and thus,

$$C_S^{\leftarrow}(6) = \underbrace{\frac{N_s(6)}{\omega_S(10,6)d(10,6)}}_{=\frac{5}{1.4}} + \underbrace{\frac{N_s(6)}{\omega_S(7,6)d(7,6)}}_{=\frac{5}{1.3}} + \underbrace{\frac{N_s(6)}{\omega_S(8,6)d(8,6)}}_{=\frac{5}{1.2}} = 3.25$$

Closeness variant $C_{S'}$ This variant is similar to C_S , however, in this variant, the requirement for a flow between a node pair is defined less strictly: here, in order to consider a node pair (s, t) , it is sufficient that at least one observed trajectory *contains* s and t .

Definition 4.8: Closeness variant $C_{S'}$

We define

$$C_{S'}^{\leftarrow}(v) = \sum_{w \in V_{S'}(v)} \frac{N_{S'}(v)}{\omega_{S'}(w, v)d(w, v)} \quad (4.7)$$

with the weight function

$$\omega_{S'}(w, v) = \begin{cases} 1 & \text{if } \exists P \in \mathcal{P} : P = (\dots, w \dots, v, \dots), v \neq w \\ 0 & \text{otherwise} \end{cases}$$

and the normalization factor

$$N_{S'}(v) = |\{w \in V \mid \exists P \in \mathcal{P} : P = (\dots, w \dots, v, \dots), v \neq w\}|$$

and the node subset $V_{S'}(v) = \{w \in V \mid w \neq v, \omega_{S'}(w, v) \neq 0\}$.

This relaxation makes this flow-based variant more like the standard closeness centrality: As in standard closeness centrality, all intermediate nodes between w and v also contribute to the closeness value.

In the example in Figure 4.3, this change raises the values of all nodes that are at least contained

in one path from \mathcal{P} , for example for node 3, we obtain $V_{S'}(3) = \{10, 1, 2, 7, 8\}$

$$\begin{aligned} C_{S'}^+(3) &= \underbrace{\frac{N_{S'}(3)}{\omega_{S'}(10,3)d(10,3)}}_{=\frac{5}{1 \cdot 2}} + \underbrace{\frac{N_{S'}(3)}{\omega_{S'}(1,3)d(1,3)}}_{=\frac{5}{1 \cdot 1}} + \underbrace{\frac{N_{S'}(3)}{\omega_{S'}(2,3)d(2,3)}}_{=\frac{5}{1 \cdot 1}} \\ &+ \underbrace{\frac{N_{S'}(3)}{\omega_{S'}(7,3)d(7,3)}}_{=\frac{5}{1 \cdot 1}} + \underbrace{\frac{N_{S'}(3)}{\omega_{S'}(8,3)d(8,3)}}_{=\frac{5}{1 \cdot 2}} = 20 \end{aligned}$$

Closeness variant C_{SW} Like the previous two variants, this variant still considers the length of the shortest paths, but distances from nodes from which there is more real-world flow, will have a larger impact on the measure value. We thus chose a weight for (w, v) proportional to the number of process trajectories between w and v .

Definition 4.9: Closeness variant C_{SW}

We define

$$C_{SW}^+(v) = \sum_{w \in V_{SW}(v)} \frac{N_{SW}(w, v)}{\omega_{SW}(w, v)d(w, v)} \quad (4.8)$$

with

$$\omega_{SW}(w, v) = |\{P \in \mathcal{P} | P = (\dots, w, \dots, v, \dots)\}|$$

and the normalization factor

$$N_{SW}(v) = \sum_{w \in V} \omega_{SW}(w, v)$$

and $V_{SW}(v) = \{w \in V | w \neq v, \omega_{SW}(w, v) \neq 0\}$

For the example shown in Figure 4.3, this yields for node 4, $V_{SW}(4) = \{10, 1, 2, 7, 3, 8, 9\}$ and thus,

$$\begin{aligned} C_{SW}^+(4) &= \underbrace{\frac{N_{SW}(4)}{\omega_{SW}d(10,4)}}_{=\frac{22}{1 \cdot 3}} + \underbrace{\frac{N_{SW}(4)}{\omega_{SW}d(1,4)}}_{=\frac{22}{2 \cdot 2}} + \underbrace{\frac{N_{SW}(4)}{\omega_{SW}d(2,4)}}_{=\frac{22}{4 \cdot 2}} + \underbrace{\frac{N_{SW}(4)}{\omega_{SW}d(7,4)}}_{=\frac{22}{4 \cdot 2}} \\ &+ \underbrace{\frac{N_{SW}(4)}{\omega_{SW}d(3,4)}}_{=\frac{22}{6 \cdot 1}} + \underbrace{\frac{N_{SW}(4)}{\omega_{SW}d(8,4)}}_{=\frac{22}{1 \cdot 1}} + \underbrace{\frac{N_{SW}(4)}{\omega_{SW}d(9,4)}}_{=\frac{22}{4 \cdot 1}} = 49.5 \end{aligned}$$

Closeness variant C_R In this and the following two variants, the assumption of shortest paths is dropped and instead the "real path length" is considered. The idea is that if a network process tends to use detours instead of the shortest path, the closeness centrality should consider the length of the actually taken paths instead of the length of the (not used) shortest path. If a network process is moving along shortest paths, the variants will coincide. We therefore need to define a flow-based path length $d^P(v, w)$. For a single trajectory $P \in \mathcal{P}$, we define the P -distance from node v to node w , denoted by $d^P(v, w)$, as the sum of the edge weights contained in P between the occurrence of v and w in P^2 , it is not defined if P does not contain v or w . It is possible that there

²To be precise: if v and w are contained multiple times in P , the minimum sum of edge weights of the edges between v and w in P is used.

are multiple $P \in \mathcal{P}$ containing both nodes v and w . For a flow-based distance $d^{\mathcal{P}}$, the single P -distances are aggregated by averaging them. Consider the graph and the trajectory set \mathcal{P} shown in Figure 4.3. Nodes 3 and 6 have a graph-theoretic distance $d(3, 6) = 2$, so we obtain the following P -distances for them: $d^{P_1}(3, 6) = 2$, $d^{P_2}(3, 6) = 2$, $d^{P_3}(3, 6) = 3$, $d^{P_4}(3, 6) = 3$, $d^{P_5}(3, 6) = 3$, while $d^{P_6}(3, 6)$ is not defined. By averaging those (defined) single values, we obtain a flow-based distance of $d^{\mathcal{P}}(3, 6) = \frac{13}{5} = 2.6$.

The flow-based distance is then used for defining variant C_R .

Definition 4.10: Closeness variant C_R

We define

$$C_R^{\leftarrow}(v) = \sum_{w \in V_R(v)} \frac{N_R(v)}{\omega_R(w, v) d^{\mathcal{P}}(w, v)} \quad (4.9)$$

with

$$\omega_R(w, v) = \begin{cases} 1 & \text{if } \exists P \in \mathcal{P} : P = (\dots, w, \dots, v, \dots), v \neq w \\ 0 & \text{otherwise} \end{cases}$$

and

$$N_R(v) = |\{w \in V \mid \exists P \in \mathcal{P} : P = (\dots w \dots v \dots)\}|$$

and $V_R(v) = \{w \in V \mid w \neq v, \omega_R(w, v) \neq 0\}$

Since for nodes v, w which do not occur in any common trajectory, there is no flow-based distance, for computing the closeness for v , only those nodes co-occurring with v in at least one trajectory can contribute to the value. Thus, the assumption of the existence of flow from each node to v cannot be upheld completely.

In the example in Figure 4.3, the following flow-based distances need to be computed in order to compute $C_R^{\leftarrow}(3)$:

$$\begin{aligned} d^{\mathcal{P}}(10, 3) &= \frac{3}{1}, & d^{\mathcal{P}}(1, 3) &= \frac{2+2}{2} = 2, & d^{\mathcal{P}}(2, 3) &= \frac{1+1+1}{3} = 1 \\ d^{\mathcal{P}}(5, 3) &\text{undefined}, & d^{\mathcal{P}}(7, 3) &= \frac{1+2+1+2}{4} = 1.5, & d^{\mathcal{P}}(8, 3) &= \frac{3}{1} \\ d^{\mathcal{P}}(4, 3) &\text{undefined}, & d^{\mathcal{P}}(9, 3) &\text{undefined}, & d^{\mathcal{P}}(6, 3) &\text{undefined} \end{aligned}$$

For node 3, we get $V_R(3) = \{10, 1, 2, 7, 8\}$ and can then compute the value for node 3 as follows

$$\begin{aligned} C_R^{\leftarrow}(3) &= \underbrace{\frac{N_R(3)}{\omega_R(10, 3) d^{\mathcal{P}}(10, 3)}}_{\frac{5}{1 \cdot 3}} + \underbrace{\frac{N_R(3)}{\omega_R(1, 3) d^{\mathcal{P}}(1, 3)}}_{\frac{5}{1 \cdot 2}} + \underbrace{\frac{N_R(3)}{\omega_R(2, 3) d^{\mathcal{P}}(2, 3)}}_{\frac{5}{1 \cdot 1}} \\ &+ \underbrace{\frac{N_R(3)}{\omega_R(7, 3) d^{\mathcal{P}}(7, 3)}}_{\frac{5}{1 \cdot 1.5}} + \underbrace{\frac{N_R(3)}{\omega_R(8, 3) d^{\mathcal{P}}(8, 3)}}_{\frac{5}{1 \cdot 3}} = 14.17 \end{aligned}$$

Closeness variant C_{RW} Similarly as before, we additionally introduce a weight function proportional to the amount of flow between two nodes. Thus, distances between nodes between which there is a large amount of flow contribute more to the measure value than distances of node pairs between which there is (almost) no flow.

Definition 4.11: Closeness variant C_{RW}

We define

$$C_{RW}^{\leftarrow}(v) = \sum_{w \in V_{RW}(v)} \frac{N_{RW}(v)}{\omega_{RW}(w, v) d^P(w, v)} \quad (4.10)$$

with

$$\omega_{RW}(w, v) = |\{P \in \mathcal{P} | s(P) = w, t(P) = v\}|$$

and the normalization factor

$$N_{RW}(v) = \sum_{w \in V} \omega_{RW}(w, v)$$

and $V_{RW}(v) = \{w \in V | w \neq v, \omega_{RW}(w, v) \neq 0\}$.

In this variant, most nodes of the example in Figure 4.3 get a value of 0, since for the example paths, only nodes 6 and 4 have an incoming flow according to the strict definition used in this variant. For computing the value for node 6, the following flow-based distances need to be computed:

$$d^P(10, 6) = \frac{5}{1}, \quad d^P(7, 6) = \frac{3 + 5 + 4 + 5}{4} = 4.25, \quad d^P(8, 6) = \frac{6}{1},$$

With $V_{RW}(6) = \{10, 7, 8\}$, we get

$$C_{RW}^{\leftarrow}(6) = \underbrace{\frac{N_{RW}(6)}{\omega_{RW}(10, 6) d^P(10, 6)}}_{\frac{5}{1 \cdot 5}} + \underbrace{\frac{N_{RW}(6)}{\omega_{RW}(7, 6) d^P(7, 6)}}_{\frac{5}{3 \cdot 4.25}} + \underbrace{\frac{N_{RW}(6)}{\omega_{RW}(8, 6) d^P(8, 6)}}_{\frac{5}{1 \cdot 6}} \approx 2.23$$

Closeness variant $C_{RW'}$ Like for the variants C_S and $C_{S'}$, we loosen the restriction of "flow from v to w " in this case. In the previous variant, a trajectory P only contributes to the flow between its start node $s(P)$ and its target node $t(P)$. In this variant, we count all trajectories going through v and w for the flow between v and w .

Definition 4.12: Closeness variant $C_{RW'}$

We define

$$C_{RW'}^{\leftarrow}(v) = \sum_{w \in V_{RW'}(v)} \frac{N_{RW'}(v)}{\omega_{RW'}(w, v) d^P(w, v)} \quad (4.11)$$

with

$$\omega_{RW'}(w, v) = |\{P \in \mathcal{P} | v, w \in P, v \neq w\}|$$

and

$$N_{RW'}(v) = \sum_{w \in V} \omega_{RW'}(w, v)$$

and $V_{RW'}(v) = \{w \in V | w \neq v, \omega_{RW'}(w, v) \neq 0\}$.

For node 6 in the example in Figure 4.3, we get $V_{RW'}(6) = \{10, 1, 2, 3, 7, 8, 4, 9\}$, so we get

$$\begin{aligned}
 C_{RW'}^{\leftarrow}(6) &= \underbrace{\frac{N_{RW'}(6)}{\omega_{RW'}(10,6)d^{\mathcal{P}}(10,6)}}_{\frac{23}{1.5}} + \underbrace{\frac{N_{RW'}(6)}{\omega_{RW'}(1,6)d^{\mathcal{P}}(1,6)}}_{\frac{23}{1.4}} + \underbrace{\frac{N_{RW'}(6)}{\omega_{RW'}(2,6)d^{\mathcal{P}}(2,6)}}_{\frac{23}{3.3.5}} \\
 &+ \underbrace{\frac{N_{RW'}(6)}{\omega_{RW'}(3,6)d^{\mathcal{P}}(3,6)}}_{\frac{23}{5.2.6}} + \underbrace{\frac{N_{RW'}(6)}{\omega_{RW'}(7,6)d^{\mathcal{P}}(7,6)}}_{\frac{23}{4.4.25}} + \underbrace{\frac{N_{RW'}(6)}{\omega_{RW'}(8,6)d^{\mathcal{P}}(8,6)}}_{\frac{23}{1.6}} \\
 &+ \underbrace{\frac{N_{RW'}(6)}{\omega_{RW'}(4,6)d^{\mathcal{P}}(4,6)}}_{\frac{23}{5.1}} + \underbrace{\frac{N_{RW'}(6)}{\omega_{RW'}(9,6)d^{\mathcal{P}}(9,6)}}_{\frac{23}{3.2}} \approx 27.9
 \end{aligned}$$

with these flow-based distances

$$\begin{aligned}
 d^{\mathcal{P}}(1,6) &= \frac{4+4}{2} = 4, & d^{\mathcal{P}}(2,6) &= \frac{3+4+3+4}{4} = 3.5, & d^{\mathcal{P}}(5,6) & \text{undefined,} \\
 d^{\mathcal{P}}(3,6) &= \frac{2+2+3+3+3}{5} = 2.6, & d^{\mathcal{P}}(4,6) &= \frac{1+1+1+1+1}{5} = 1, & d^{\mathcal{P}}(9,6) &= \frac{2+2+2}{3} = 2
 \end{aligned}$$

4.3 Comparison methods for rankings

Given a graph $G = (V, E)$ and a set of walks in G , denoted by \mathcal{P} , all introduced measures can be computed (see Section 4.4 for a description of which datasets will be used as G and \mathcal{P}). Thus, for each measure and each dataset, each node $v \in V$ is assigned a value. In order to analyze the obtained data, this section will introduce various methods for comparing values and their resulting rankings.

Rankings For a measure $c: V \rightarrow \mathbb{R}$, with $V = \{v_1, v_2, \dots, v_n\}$, we say that c induces an order \leq_c on V by:

$$v_i \leq_c v_j \Leftrightarrow c(v_i) \geq c(v_j).$$

When the elements of V are written in a sequence $(v_{i_1}, v_{i_2}, \dots, v_{i_n})$ where $\{v_{i_1}, \dots, v_{i_n}\} = V$ such that $v_{i_j} \leq_c v_{i_{j'}}$ for any $j, j' \in \{1, \dots, n\}$, we can deduce a ranking $\sigma: V \rightarrow \{1, \dots, n\}$ by assigning each element of V its position number in the ordered sequence, hence the element of V with the highest value of c , here i_1 , gets a ranking value $\sigma(i_1) = 1$ and so forth. We say that elements with a low ranking value $(1, 2, 3, \dots)$ are *ranked high* while elements with a high ranking value $(n, n-1, \dots)$ are *ranked low*. There are several strategies for handling ties, i.e., elements of V with the same measure value c . Ordinal ranking σ^o assigns distinct numbers to each element, where the positioning of elements with the same measure value c can be done according to some arbitrary criterion, for example randomly, which yields random ranking σ^r . Standard competition ranking σ^{min} can assign the same ranking number to different elements of V : Elements with an equal measure value c are all assigned the minimal position of the elements in the ordered sequence. Fractional ranking σ^f assigns the mean position number of the tie elements' position numbers. An example is given in Table 4.3.

Given then two measures c and d on V , and their corresponding rankings σ_c and σ_d , different approaches exist for comparing them.

Table 4.3 Let V be a set of items and $c(v)$ a measure inducing an order on V . σ^o shows the deduced ordinal ranking, σ^{min} the deduced standard competition ranking, and σ^f the fractional ranking, σ^r (one possible instantiation of) random ranking.

V	$c(v)$	Sequence	Ranking methods			
			σ^o	σ^{min}	σ^f	σ^r
A	10	B	1. B	1. B	1. B	1. B
B	30	F	2. F	2. F	2. F	2. F
C	5	A	3. A	3. A	4. E	3. A
D	10	D	4. D	3. D	4. A	4. E
E	10	E	5. E	3. E	4. D	5. D
F	25	C	6. C	6. C	6. C	6. C

Pearson correlation of measures The Pearson correlation coefficient is a measure quantifying the strength of a linear relationship between two variables. Let \vec{c} and \vec{d} denote vectors containing the values $c(v_i)$ and $d(v_i)$ for each $v_i \in V$. Let $\vec{c}_i := c(v_i)$ ($\vec{d}_i := d(v_i)$) respectively, denote the value of c (respectively d) for node v_i . Then, $\bar{c} = \frac{1}{n} \sum_{i=1}^n c(v_i)$ is the mean value of \vec{c} (and defined correspondingly for d). The Pearson correlation coefficient [Gal89, Bra46] is then defined as

$$\tau_P(\vec{c}, \vec{d}) = \frac{\sum_{i=1}^n (\vec{c}_i - \bar{c})(\vec{d}_i - \bar{d})}{\sqrt{\sum_{i=1}^n (\vec{c}_i - \bar{c})^2 \sum_{i=1}^n (\vec{d}_i - \bar{d})^2}} \quad (4.12)$$

The Pearson correlation coefficient ranges from 1 (indicating a perfect linear relationship) over 0 (no linear relationship) to -1 (indicating a perfect negative linear relationship). This measure is widely used. However, since the measures introduced in Section 4.2 are not necessarily related linearly (the following sections will show that they are related, but rarely in a purely linear way), the Pearson correlation coefficient is not the appropriate measure for comparing the measure values.

Spearman's rank correlation coefficient If the values of two measures are related, but not linearly, the Pearson correlation coefficient is not a good measure for quantifying the relationship. An alternative possibility is given by Spearman's rank correlation coefficient [Spe04]. Instead of testing for a linear relationship, Spearman's rank correlation coefficient tests for any monotonic relationship between the values of the two measures. This is done by replacing the measure values with their ranking position and computing the Pearson correlation coefficient on the rank variables. Using this approach, non-linear relationships can also be detected, i.e., any monotonic relationship can be detected, and the measure is more robust against outliers than the Pearson correlation coefficient.

Ranking deviations and span of ranking positions Basic metrics for comparing two rankings σ_c and σ_d are the absolute differences of the ranking positions for each node $v \in V$, i.e., $\Delta_{cd}(v) = |\sigma_c(v) - \sigma_d(v)|$ for a $v \in V$. For a first comparison of two rankings, the minimal, maximal, and mean value of all Δ -values might be of interest. For comparing k rankings, $\sigma_1, \dots, \sigma_k$, this can be generalized yielding a measure we call the span of ranking positions of a node v , and is computed by

$$span(v) = \max_{i,j \in \{1, \dots, k\}} \Delta_{ij}(v) = \max_{i=1, \dots, k} \sigma_i(v) - \min_{i=1, \dots, k} \sigma_i(v). \quad (4.13)$$

Also for the span of ranking positions, the minimal, maximal, and mean values over all $span$ -values can provide insights about the similarity of the rankings.

Kendall rank correlation coefficient A popular approach for comparing two rankings is the Kendall rank correlation coefficient introduced by [Ken38]. For this measure, all pairs of elements

in each ranking are considered: A pair (v_i, v_j) is called *concordant* if their order is consistent in both rankings, more precisely, $\sigma_c(v_i) < \sigma_c(v_j) \Leftrightarrow \sigma_d(v_i) < \sigma_d(v_j)$. A pair (v_i, v_j) is called *discordant* if $\sigma_c(v_i) < \sigma_c(v_j) \Leftrightarrow \sigma_d(v_i) > \sigma_d(v_j)$. A pair is called a *tie in σ_c* if $\sigma_c(v_i) = \sigma_c(v_j)$ and $\sigma_d(v_i) \neq \sigma_d(v_j)$ (a node pair being a tie in σ_2 is defined accordingly). A node pair (v_i, v_j) is called a *tie in σ_c and σ_d* if $\sigma_c(v_i) = \sigma_c(v_j)$ and $\sigma_d(v_i) = \sigma_d(v_j)$. Let C then be the number of concordant pairs, D the number of discordant pairs, and T_c (T_d) the number of tie pairs in σ_c (σ_d).

Then the Kendall rank correlation coefficient is defined as

$$\tau_K(\sigma_c, \sigma_d) = \frac{C - D}{\sqrt{(C + D + T_c) \cdot (C + D + T_d)}} \quad (4.14)$$

which, in the case of no ties, can be simplified to $\tau_K = \frac{C-D}{C+D}$. In both cases, it holds that $-1 \leq \tau_K \leq 1$. Furthermore, if there are more concordant pairs than discordant pairs, then $\tau_K > 0$.

While τ_K is a well-defined measure, satisfying coincidence (at least with obvious modifications), symmetry, and the triangle inequality, it does exhibit some undesired behavior in some contexts: When considering rankings, we are mostly interested in the highly ranked elements and less interested in the low-ranked elements. Therefore, we might pay less attention to the exact position of low-ranked elements as long as they are low-ranked in both rankings, than to the difference of the ranking positions in the top k positions. This, however, is not taken into account in the computation of the Kendall coefficient (nor in that of any other measure introduced in the previous paragraphs). Consider the following three rankings (written as a sequence)

$$\begin{aligned} \sigma_1 &= (1, \dots, 100), \\ \sigma_2 &= (1, \dots, 90, 100, 99, \dots, 91) \text{ and} \\ \sigma_3 &= (10, 9, \dots, 1, 11, 12, \dots, 100). \end{aligned}$$

Comparing σ_1 and σ_2 with the Kendall rank correlation coefficient yields $D = \binom{10}{2} = 45$ discordant pairs and $C = \binom{90}{2} + 90 \cdot 10 = 4905$ concordant pairs, and hence (since there are no ties) $\tau_K(\sigma_1, \sigma_2) = 4860/4950 = 0.98$, the same as for σ_1 and σ_3 . This behavior might not be desired when comparing the rankings induced from centrality measures. In this case, it might be advantageous to put more emphasis on similar ranking behaviors among the top-ranked elements than among the least highly ranked elements. Furthermore, the *exact* ranking positions of a node are often of less interest than whether it is ranked high or low by both rankings. For these reasons, we propose a measure called *Weighted overlap of rankings* which will be introduced in the following paragraph.

Weighted overlap of rankings For the two reasons stated above and for want of an interpretable and illustrative measure, we developed the weighted overlap of rankings for comparing rankings. It is based on the idea that the top x nodes of both rankings do not need to be in perfect order to get a high value, but should at least contain nearly the same elements. We therefore consider the overlap of the top x elements, i.e., the number of elements that are among the x highest ranked elements in both rankings,

$$ov(\sigma_c, \sigma_d, x) = |\{v \in V | \sigma_c(v) \leq x\} \cap \{v \in V | \sigma_d(v) \leq x\}|. \quad (4.15)$$

In the following, we assume the rankings to be free of ties.

Obviously, for any two rankings σ_c and σ_d , it holds that $ov(\sigma_c, \sigma_d, 0) = 0$ and $ov(\sigma_c, \sigma_c, n) = n$. Note that $ov(\sigma_c, \sigma_d, x) = x$ for a $x \in \{0, \dots, n\}$ does not imply equality of the two rankings, also not on the first x positions. If, however, $ov(\sigma_c, \sigma_d, x) = x$ holds for all x , it implies $\sigma_c = \sigma_d$.

We use this definition to define a preliminary measure for comparing two rankings, from which we will derive the weighted overlap of rankings. We refer to the preliminary version as unweighted

overlap of rankings and define it as

$$\tau = \frac{4}{n^2} \sum_{x=0}^n (x - ov(\sigma_c, \sigma_d, x)) \quad (4.16)$$

The differences between each possible x of the actual overlap and the perfect overlap (the overlap of identical rankings which is x) are computed and summed up. A graphical explanation is: If the overlap function is plotted as a function of x , τ corresponds to the area between the overlap line and the identity line. The identity line is the overlap curve of two identical rankings (see Figure 4.4, where the value of τ corresponds to the red area).

The total area between the overlap and the identity function is normalized by its maximal value $\frac{n^2}{4}$. That this is the maximal possible area that can be reached by rankings can be seen by a graphical argument as well: It holds that the slope of the overlap line for any two rankings cannot exceed 2. To prove this claim, consider the difference in overlap from any x to $x+1$. For any two rankings σ_c and σ_d , it is clear that from any x to $x+1$, the overlap cannot decrease, i.e., $ov(\sigma_c, \sigma_d, x) \leq ov(\sigma_c, \sigma_d, x+1)$ for all $0 \leq x \leq n-1$. Furthermore, from any x to $x+1$, the overlap can increase by at most 2 because exactly two elements are considered³ when computing $ov(\sigma_c, \sigma_d, x+1)$, namely $\sigma_c(x+1)$ and $\sigma_d(x+1)$. There are four cases to distinguish: (i) If the two considered elements are the same, $\sigma_c(x+1) = \sigma_d(x+1)$, the overlap increases by 1. (ii) If they are not the same, and $\sigma_c(x+1)$ is contained in the first x entries of ranking σ_d , and vice versa (i.e., $\sigma_d(x+1)$ is contained in the first x entries of σ_c), the overlap increases by 2 since both considered elements increase the intersection. (iii) If only one of the two considered elements is contained in the first x entries of the other ranking, i.e., $\sigma_c(x)$ is contained in first x entries of σ_d or vice versa, the overlap increases by 1. (iv) If neither of the two considered elements on position $x+1$ is contained in the first x positions of the other ranking, the overlap does not change. Hence, it holds that

$$0 \leq ov(\sigma_c, \sigma_d, x+1) - ov(\sigma_c, \sigma_d, x) \leq 2$$

which implies that the slope of the overlap function cannot exceed 2.

Together with the facts that $ov(\sigma_c, \sigma_d, n) = n$, $ov(\sigma_c, \sigma_d, 0) = 0$, and $ov(\sigma_1, \sigma_2, x) \leq x$ for all $x \in \{0, \dots, n\}$, it follows that for any two rankings σ_c and σ_d , it holds that

$$\begin{aligned} 0 \leq x \leq \lfloor \frac{n}{2} \rfloor &\Rightarrow ov(\sigma_c, \sigma_d, x) \geq 0 \\ x \geq \frac{n}{2} &\Rightarrow ov(\sigma_c, \sigma_c, x) \geq 2x - n \end{aligned}$$

Exactly this overlap line is realized by the rankings $\sigma_c = (1, \dots, n)$ and its reverse $\sigma_d = (n, \dots, 1)$. A larger area is not possible since the overlap line needs to reach the point (n, n) , but cannot be steeper than 2. This maximal possible area between the overlap line and the identity line is indicated by the gray area in Fig. 4.4. Since this maximal area has the size $n^2/4$, τ is normalized by this factor, yielding a measure between 0 and 1.

The measure τ satisfies symmetry, coincidence, and the triangle inequality. At the same time, the graphic visualization makes it easy to understand the two rankings of interest, e.g., whether the two rankings differ with respect to the highly ranked elements or to the lower ranked elements.

τ , however, still exhibits the same behavior as the Kendall rank correlation: Swaps in ranking positions have the same impact, regardless of whether they occur for high-ranked elements or for lower-ranked elements. For this reason, we modify it by introducing a weight where the difference from the perfect overlap for each x is weighted dependent on x . We chose a weight decreasing

³Here, the assumption of no ties is needed.

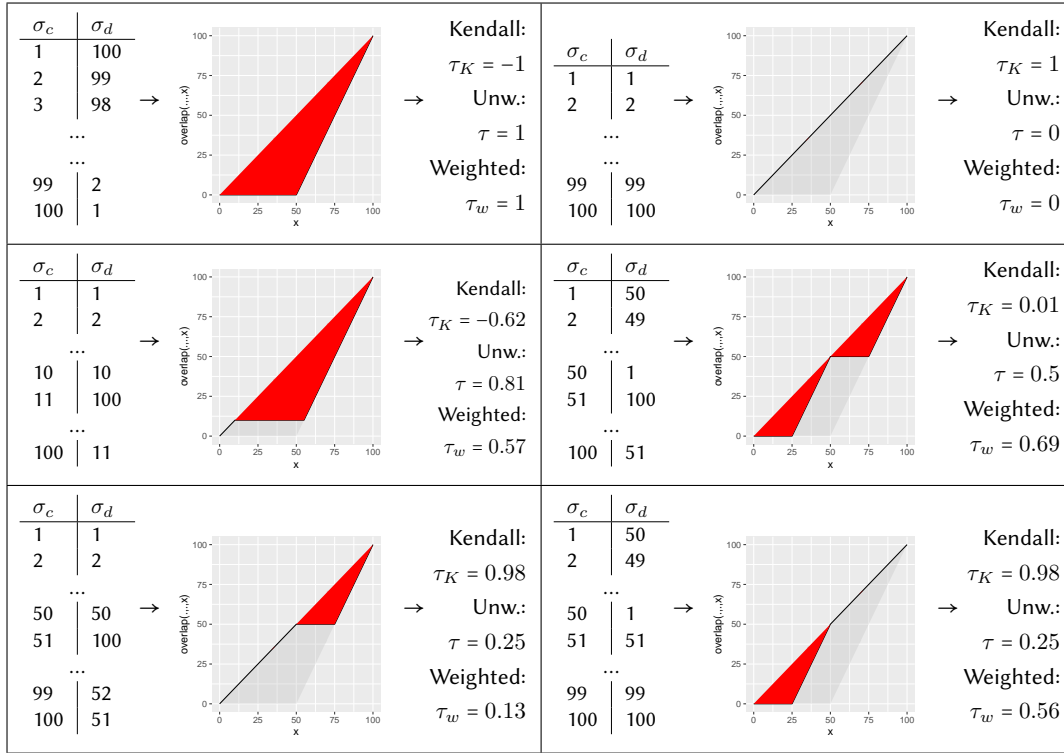


Figure 4.4 Graphical explanation of the introduced overlap measures τ and τ_w : When plotting the overlap of two rankings σ_c and σ_d as a function of x , for $x \in [0, n]$, the measure τ gives the area between the identity line and the overlap function (red area), normalized by its maximal value (gray area). The bottom panels illustrate that the Kendall rank correlation τ_K and the overlap measure τ are not affected by the *position* of ranking differences: Differences in ranking positions between σ_c and σ_d have the same effect whether they occur in the top x or bottom x elements. This undesired behavior is modified by the introduction of a weight that penalizes differences in rankings more when they occur in the top ranked elements (yielding a weighted overlap measure τ_w).

linearly with x , but other variants might also be appropriate. There is still another counterintuitive behavior: consider the maximally reachable value of $x - ov(\sigma_c, \sigma_d, x)$ for each x . It is easy to see that for $x \leq n/2$, each summand $x - ov(\sigma_c, \sigma_d, x)$ can contribute at most x to the sum, for $x = n/2$, $x - ov(\sigma_c, \sigma_d, x)$ can contribute the largest value, i.e., $n/2$, and for values of $x > n/2$, each summand can contribute at most $n - x$ to the sum. This is why we normalize each summand by its maximal value (introducing a normalization factor η . We then get

$$\tau_w = \frac{2}{n(n-1)} \sum_{x=1}^n w(x) \cdot \frac{x - ov(\sigma_c, \sigma_d, x)}{\eta(x)} \quad (4.17)$$

with $w(x) = n - x$ and

$$\eta(x) = \begin{cases} x & x \leq \lfloor n/2 \rfloor \\ n - x & \text{otherwise} \end{cases} \quad (4.18)$$

Using these adaptations, we gain a measure that returns 0 for identical rankings and 1 for a ranking and its reverse. For other rankings, τ_w penalizes differences in the rankings more if they occur at higher ranking positions (1, 2, 3, ...) than if they occur at lower ranking positions ($n, n-1, \dots$) (see also Figure 4.4 for examples).

Table 4.4 Properties of the datasets used. $|V|$ and $|E|$ denote the cardinality of the node and edge set of the underlying graph, $|\mathcal{P}|$ the number of observed trajectories.

Dataset	$ V $	$ E $	$ \mathcal{P} $	Path length	
				Range	Average
DB1B	415	5141	86m	[1, 12]	1.4
LT (Lines)	268	626	4.8m	[2, 49]	8.2
LT (Transitive)	268	13172	4.8m	[2, 49]	8.2
Wiki	4589	119804	51261	[1, 82]	5
Wordmorph	1008	8320	11651	[3, 55]	5.0

Other measures There also exist other measures that we will not use in this work: One possible way to compare two rankings is to use edit distance, which was originally used for comparing strings [Dam64]. In general, for edit distances, several edit operations are defined, usually insertions, deletions, substitutions, and swapping of symbols. All these operations are associated with costs. The edit distance between two strings is then the minimum cost for transforming one string into another using the allowed edit operations. Hannak et al. [HSK⁺13] for example used an edit distance that counts the number of elements that need to be inserted, deleted, substituted or swapped in order to transform one list into another to compare rankings from search engine results.

In the next section, we will describe the datasets on which the previously introduced flow-based centralities were applied before presenting the results in the subsequent sections.

4.4 Datasets used

For all introduced measures, a dataset containing a network structure and (real or synthetic) network flow is needed. Since we mainly consider closeness and betweenness centrality, only datasets containing flow with a transfer mechanism are appropriate. Furthermore, each single entity of the network flow should have a target that it tries to reach as fast as possible. Otherwise, if there is no incentive for the entities to use the shortest path (or if they do not have any target at all), it makes no sense to compare these real-world flow processes with process models assuming shortest paths. For network processes that do not have a target or do not use shortest paths, the application of centrality measures assuming shortest paths is questionable anyway. For this reason, the following datasets were used (see Table 4.5 for an overview of the datasets used, and Table 4.4 for the basic properties of these datasets).

Airline transportation (DB1B) We used a dataset containing passenger journeys in the air transportation system in the US, as described in Chapter 3. Here, we set a threshold for the insertion of nodes and edges into the network as follows: A node was only inserted if it was contained in at least 100 passenger itineraries, and an edge from node v to node w was inserted if there existed at least ten passenger itineraries with a flight from an airport in the city represented by v to an airport in the city represented by w . This procedure yielded a network containing 415 nodes. Since for almost all edges (v, w) , it holds that the edge (w, v) also exists, we modeled the network as an undirected network (and include the undirected edge (v, w) if the directed edge (v, w) or (w, v) exists). This yielded 5141 undirected edges.

London Transport As a second network flow from the domain of passenger transportation, we used a dataset provided by Transport for London [Tra17] (described in Chapter 3), containing passenger journeys within the London transportation system. Here, we used the timetables to construct two different versions of the transport network, for the following reason. There

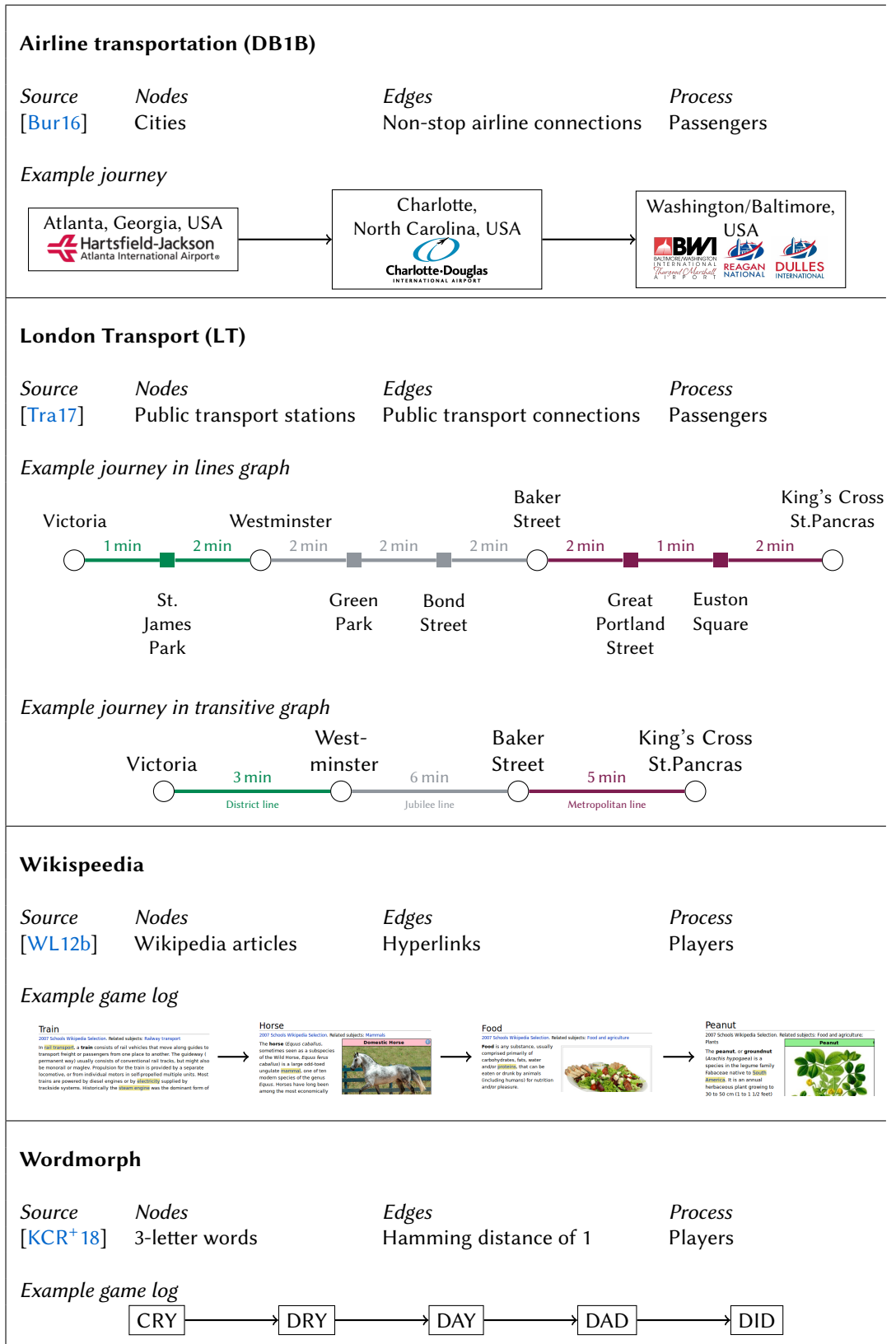
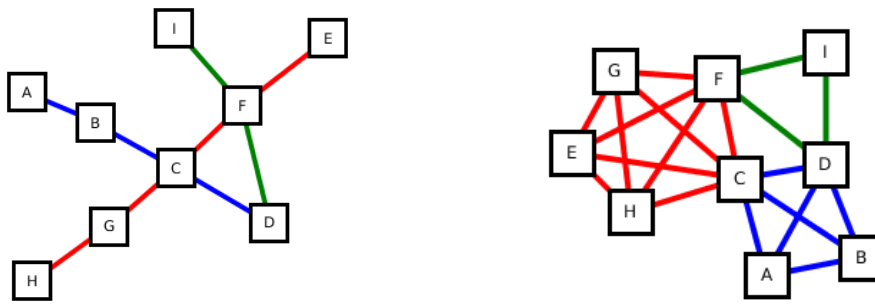


Figure 4.5 Overview of the datasets used.

are two different perspectives on the system: the perspective of a single passenger and the perspective of an administrator of the system. From the perspective of a single passenger, when getting on a train at station A and getting off the train at station B, the intermediate stations through which the train passed, are not of relevance for the passenger. For the passenger, the relevant network nodes are those in which they change trains. For this reason, we use the network representation as constructed for the analysis in Chapter 3, where each node represents a station, and there is an edge from v to w if there is a train connection from station v to w without train changes. The edges were weighted by travel time extracted from the time tables (if there are several possible connections between two stations, the smaller travel time was chosen). Thus, when a passenger journey is modeled in this kind of network, the corresponding walk only contains stations where trains are changed—intermediate stops of the *train* (where the passenger does not get off the train) are not included in the passenger journey. This type of network, however, is unable to show the physical structure of the system: When a train moves from one station to another, it needs to pass the intermediate stations in between. For this reason, we introduced a second type of network where a node again represents a station, and there is an edge from station v to station w if there is a train connection from v to w *without intermediate stops*. This type of graph rather resembles the route map of the Underground system (which is why we call this network type *lines graph*), while the first type of network can be understood as the transitive closures of the lines graph (separately for each single Underground line) which is why we call this network type *transitive graph*. Figure 4.6 shows an example of how the same transportation system consisting of three different train lines leads to two different network representations. When considering centralities, both representations can be meaningful depending on the application scenario: If one is interested in the most central stations with respect to the number of passengers that (hypothetically) pass through a station in order to change platforms, the transitive graph might be the better representation. On the other hand, if one is interested in central nodes with respect to the amount of flow (number of passengers) traveling through the station in trains, the lines graph is a better representation. In both cases, this procedure yielded a graph with 268 nodes, once with 626 edges (for the lines graph), once with 13172 edges. Both are connected.

Wikipedia A different type of network flow processes using (or at least aiming at taking) shortest paths is provided by West and Leskovec [WPP09, WL12b] containing human navigation in information networks (see Chapter 3 for a detailed description of this dataset). Here, we furthermore excluded solutions longer than 30 steps (for a note on how this decision affected the results, see Section 4.7. This yielded more than 50000 solutions navigating in a network of 4589 nodes and almost 120000 edges.

Wordmorph We used a second dataset containing game logs, from people playing the game Wordmorph. In this game, a player is given two words of the same length, and needs to transform the one word into the other by changing only one letter at a time—and each letter transformation needs to yield an existing word. A valid sequence of transformations is for example cry→dry→day→dad→did. We used a dataset collected by Kőrösi et al. [KCR⁺18] who developed an app in which people can play this game while game logs are collected for research purposes. In this app, players can choose whether they want to play with English or Hungarian words and which length the words should have (3 to 7). We restricted our analysis to successful games with English words of length 3, which resulted in approximately 11000 game logs (out of almost 20000 game logs of the original dataset) being used for the analysis. The underlying network consists of 1008 nodes each representing a 3-letter English word (official English Scrabble words from WordFind [Wor]) and 8320 (undirected) edges where two nodes v and w are connected if their represented words have a Hamming distance of 1, i.e., can be transformed into each other by changing one letter.



(a) Lines graph of a transportation system.

(b) Transitive graph of the same transport system.

Figure 4.6 Example for two different representations of the same transportation system.

Application of flow-based measures For each dataset, all introduced flow-based centrality measures as well as the standard closeness and betweenness centrality⁴ were computed. The measure values were used to deduce a ranking for each dataset and each measure, where fractional ranking was applied.

4.5 How robust are standard centrality measures?

The first question which is going to be investigated is how robust the standard centrality measures are against violations in their process model. Chapter 3 showed that the network flows contained in the considered datasets do not satisfy the assumptions of shortest paths and equal amount of flow between each node pair. In this section, by comparing the rankings of the standard centrality measures to the rankings of the flow-based measures, we can investigate whether the violations of the assumptions makes a big difference for the resulting rankings or whether the standard centrality measures are robust against those violations.

4.5.1 Correlation of measures

For this reason, we compute the Spearman rank coefficient of the values of each flow-based measure and the values of the corresponding standard measure, as well as the weighted overlap measure between the corresponding rankings⁵. For the closeness measures, we compare the in-closeness variants to the standard in-closeness centrality, and the out-closeness variants to the standard out-closeness centrality (for datasets with undirected networks, standard in- and out-closeness coincide, while the flow-based in- and out-variants are not necessarily equal). Table 4.5 shows for each dataset and each flow-based betweenness measure the Spearman rank correlation to standard betweenness centrality and the weighted overlap τ_w of the corresponding rankings. Table 4.6 shows the corresponding values for the closeness variants. Figures 4.7 and 4.8 show the measure values and the corresponding rankings of the flow-based variants compared to the values and rankings of the corresponding standard centrality value. Figure 4.7 (for betweenness measures) and 4.8 (for closeness measures) show for each variant the measure value against the value of its standard centrality measure (blue points) and its corresponding rank position (red). For each single plot, each node is hence represented by two data points (red and blue). To facilitate comparison, the values for each measure c were scaled to the interval $[0, 1]$ by using the following transformation

⁴For betweenness centrality, we used betweenness centrality including endpoints as standard betweenness centrality. This is due to the fact that all introduced flow-based variants also include the endpoints of the trajectories into the computation.

⁵Since normalization of the weighted overlap is only applicable if the ranking does not contain any ties, we computed the weighted overlap between the rankings obtained by applying random ranking. The remaining analysis was done on the rankings obtained by fractional ranking.

$c(v) \rightarrow \frac{c(v) - \min_w(c(w))}{\max_w(c(w)) - \min_w(c(w))}$. The ranking positions which normally range between 1 and $|V|$ were (only for these figures) also scaled to the interval $[0, 1]$ and reversed by $1 - rank$ such that the node with the highest measure value has a rank of 1.0.

Betweenness measures From Table 4.5, it can be seen that the correlations are high for most datasets and variants—most are above 0.7. This impression is supported by Figure 4.7 where a positive relationship between any flow-based betweenness variant and standard betweenness centrality can be seen for each dataset. This is particularly visible for the DB1B dataset, where the correlations with standard centrality are between 0.87 and 0.89 for each variant. The weighted overlap τ_w is also low, between 0.16 and 0.22 for all variants for DB1B (note that τ_w is 0 for equal rankings and 1 for reverse rankings). Such high correlations and small τ_w values could have been expected for the variant B_R since the trajectories actually taken are very close to shortest paths in this dataset. It is, however, surprising that the other measure variants also show such high correlations since in this dataset, almost 50% of all node pairs were not the source and target of any observed trajectory. Furthermore, the node pairs used as source and target of at least one trajectory were not used equally often: A few node pairs were used by a high number of trajectories, while most node pairs were used rarely. This should be visible in the values of the measure variants B_{SW} and B_{RW} in which node pairs with a high amount of flow between them contribute more to the final value than node pairs used less frequently. This is, however, not the case for the DB1B dataset: Here, all flow-based measure variants show an approximately equal correlation to the standard betweenness centrality. At the same time, Figure 4.7 reveals that especially for the variants B_{SW} , B_R , and B_{RW} , there is a small number of nodes in which the values of the flow-based measure and the standard betweenness centrality do not match: There are nodes with high standard betweenness values and low flow-based values, and vice versa.

For all other datasets, the correlations are less strong which can also be seen in Figure 4.7: A positive correlation between the measure values can be observed, but less strong than for DB1B. Especially for the game datasets, Wikispeedia and Wordmorph, it is striking that the flow-based measure variants assign (relatively) higher values to nodes that only have small standard betweenness values. From Table 4.5, it can be observed that the correlations of the flow-based measures incorporating observed trajectories instead of shortest paths, i.e., B_R and B_{RW} , are smaller than those for the variants B_S and B_{SW} . For these measure variants, the correlation drops to values of 0.49 and 0.58, respectively. This suggests that for these datasets, the violation of the assumption of shortest path does have an effect on the resulting rankings. Although the observed trajectories do not deviate much from the actual shortest paths (for London Transport, the observed trajectories are longer than the shortest paths by a factor of only 1.02 on average), there is an observable effect on the measure values.

Closeness measures For the closeness variants, the correlations with the standard closeness centrality are generally lower than the correlations for the betweenness centrality variants. Here, for the Wikispeedia dataset, there is even a (slightly) negative correlation for all flow-based out-closeness variants. For the in-closeness variants, however, the correlation is constantly between 0.74 and 0.79. For the datasets DB1B, the transitive graph of London Transport, and Wordmorph, it is particularly noticeable that the correlations and the weighted overlap coefficient τ_w are different for the different datasets, but are approximately constant across the different variants. There are also no considerable differences between the corresponding in- and out-variants. For the lines graph of London Transport, however, the correlations between all variants and the standard closeness centrality are quite high—except for those variants incorporating a weight proportional to the amount of flow, i.e., all variants with a subscripted W . For those variants, the correlation drops from 0.8 to 0.5. This effect is not present for any other dataset. Note furthermore that for each dataset and almost each variant, there is at least one node that is ranked highest by the standard closeness *and* by the flow-based variants.

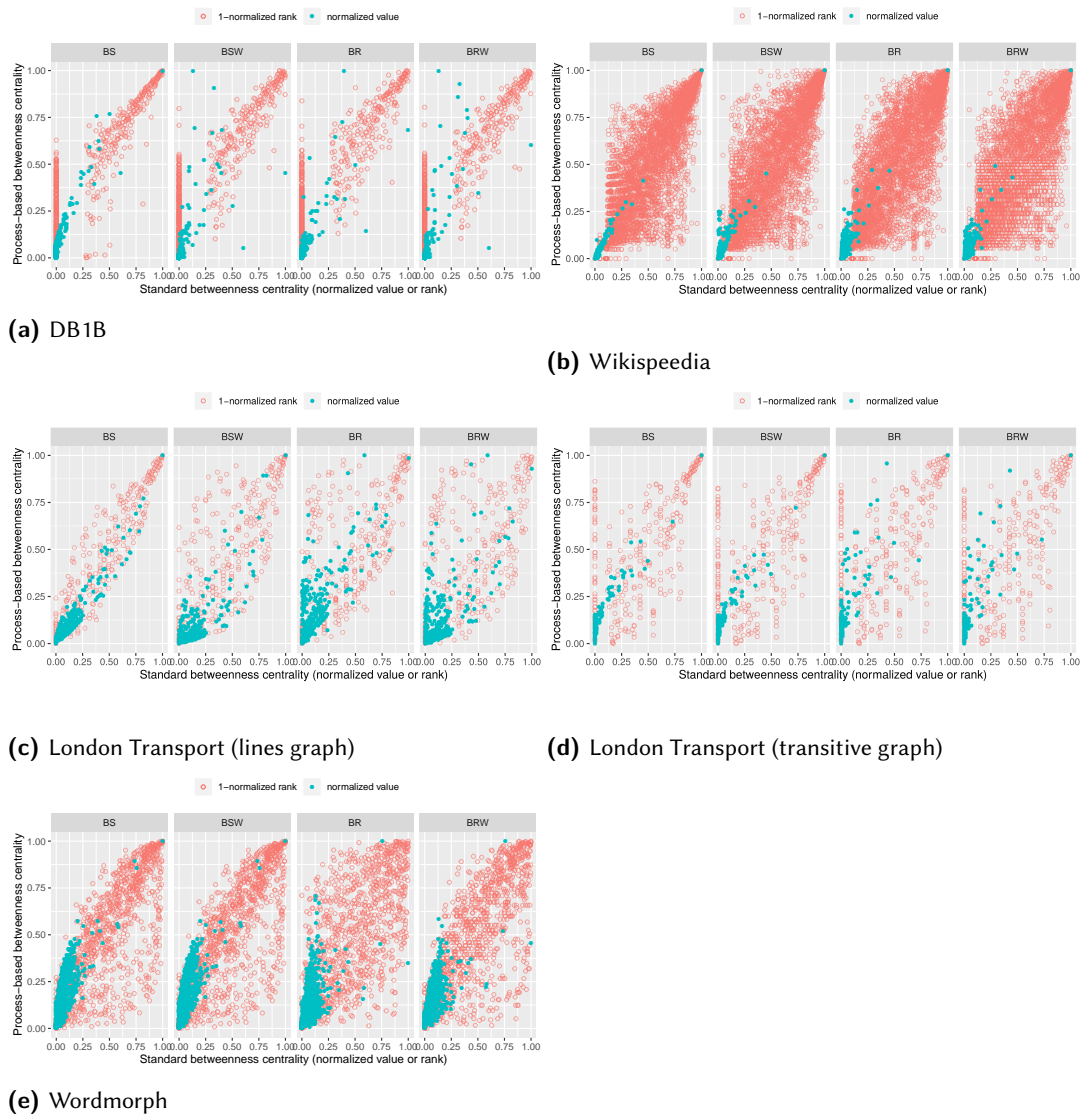


Figure 4.7 Flow-based betweenness variants compared to standard betweenness Each point represents a node of the network, its values for the (standard and flow-based) measures are drawn in blue. For the sake of better readability, the values of all measures were scaled to the interval $[0, 1]$. The figures also contain the resulting rankings of the nodes with respect to the corresponding measures. For these figures, the rankings were also scaled to the interval $[0, 1]$ and the value $1 - rank$ was drawn, such that the node with the highest measure value gets the rank 1.0 and the node with the lowest measure value gets the rank 0.0.

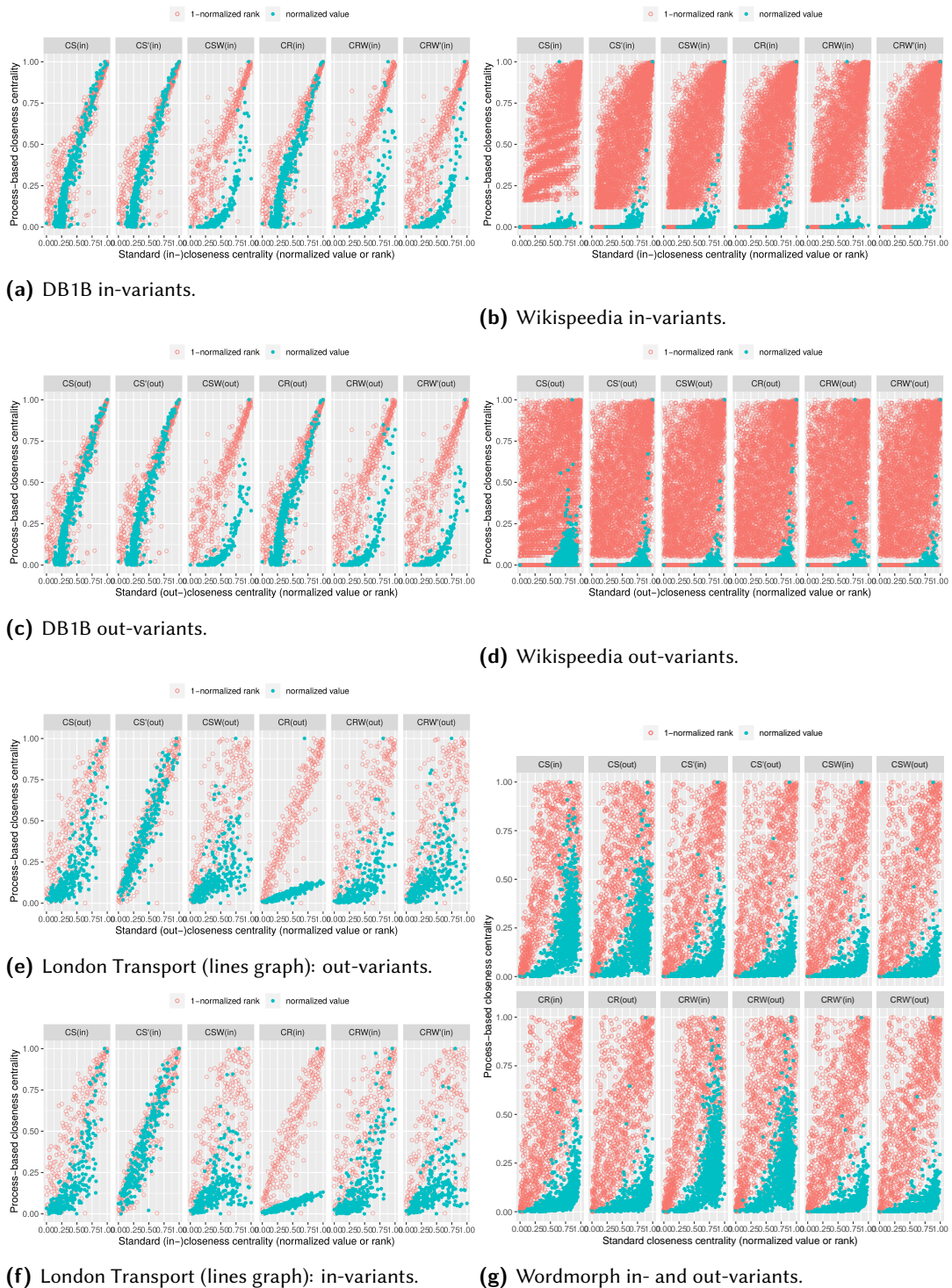


Figure 4.8 Flow-based closeness variants compared to standard closeness Each point represents a node of the network and its values for the (standard and flow-based) measures are drawn in blue. For the sake of a better readability, the values of all measures were scaled to the interval $[0, 1]$. The figures also contain the resulting rankings of the nodes with respect to the corresponding measures. For these figures, the rankings were also scaled to the interval $[0, 1]$ and the value $1 - rank$ was drawn, such that the node with the highest measure value gets the rank 1.0 and the node with the lowest measure value gets the rank 0.0. If applicable, the flow-based in-closeness variants were compared with standard in-closeness, and the flow-based out-closeness variants with standard out-closeness.

Table 4.5 Correlation between flow-based betweenness measures and standard betweenness centrality Spearman rank correlation between the flow-based betweenness centrality measures and the standard betweenness centrality and the weighted overlap of the corresponding rankings (τ_w). For Spearman correlation, starred values (*) indicate a p -value of < 0.05 . Note that the Spearman correlation coefficient ranges from -1 (negative linear correlation) via 0 (no correlation) to 1 (positive linear correlation) while τ_w ranges from 0 (equality of rankings) to 1 (reverse rankings).

Data set	Spearman Correlation			
	B_S	B_{SW}	B_R	B_{RW}
LT (Lines)	0.92*	0.72*	0.59*	0.49*
LT (Transitive)	0.62*	0.65*	0.58*	0.62*
DB1B	0.88*	0.89*	0.87*	0.88*
Wikispeedia	0.83*	0.81*	0.81*	0.79*
Wordmorph	0.75*	0.75*	0.63*	0.73*

Data set	Weighted overlap τ_w			
	B_S	B_{SW}	B_R	B_{RW}
LT (lines)	0.17	0.32	0.37	0.46
LT (transitive)	0.27	0.27	0.36	0.34
DB1B	0.17	0.23	0.22	0.23
Wikispeedia	0.21	0.26	0.30	0.31
Wordmorph	0.36	0.36	0.48	0.41

4.5.2 Deviation of rankings

The previous section showed that almost all flow-based variants show a correlation to their corresponding standard centrality measure, but the relationship is far from perfect. For each dataset and each measure variant, there are a considerable number of nodes whose importance is rated differently by standard centrality and by the flow-based variant. In order to quantify to which extent the rankings of a single node can vary among the variants, we used the span of ranking positions introduced in Section 4.3. For each node, its span of ranking positions was computed—separately for the betweenness and the closeness measures. Table 4.7 shows for each dataset, the minimal, maximal and mean values of the nodes' span of ranking positions. Although the correlations between the measure variants and the standard centrality measures appeared to be quite high in the previous section, Table 4.7 shows that there is considerable movement of ranking positions among the flow-based variants and standard centrality. Even for the DB1B dataset, where the correlations between the flow-based variants and the standard centralities seemed high, the maximum ranking difference among all variants is on average 62 positions which is 15% of all ranking positions. There is even one node with a ranking difference of 348 (out of 462 positions) among the variants. For Wikispeedia, we found an average span of ranking positions of more than 800 positions for the betweenness measures, and almost 2000 positions for the closeness measures (for $|V| = 4589$).

This analysis shows that although the flow-based variants are correlated with the standard centrality measures, there are considerable differences in the rankings for the flow-based variants which we will analyze in detail in the next sections.

Table 4.6 Correlation between flow-based closeness measures and standard closeness centrality
 Correlation between the flow-based closeness centrality measures and their corresponding standard closeness centrality measures and the weighted overlap of the corresponding rankings (τ_w). The flow-based in-(/out-)closeness was (if applicable) compared to the standard in-(/out-)closeness. For Spearman correlation, starred values (*) indicate a p -value of < 0.05 . Note that the Spearman correlation coefficient ranges from -1 (negative linear correlation) via 0 (no correlation) to 1 (positive linear correlation) while τ_w ranges from 0 (equality of rankings) to 1 (reverse rankings).

Data set	Spearman Correlation					
	C_S^{\leftarrow}	C_S^{\rightarrow}	$C_{S'}^{\leftarrow}$	$C_{S'}^{\rightarrow}$	C_{SW}^{\leftarrow}	C_{SW}^{\rightarrow}
LT (Lines)	0.84*	0.84*	0.93*	0.93*	0.64*	0.70*
LT (Transitive)	0.49*	0.47*	0.51*	0.50*	0.51*	0.49*
DB1B	0.88*	0.88*	0.88*	0.88*	0.88*	0.88*
Wikispeedia	0.76*	0.34*	0.79*	0.42*	0.79*	0.41*
Wordmorph	0.58*	0.55*	0.67*	0.64*	0.66*	0.62*
	C_R^{\leftarrow}	C_R^{\rightarrow}	C_{RW}^{\leftarrow}	C_{RW}^{\rightarrow}	$C_{RW'}^{\leftarrow}$	$C_{RW'}^{\rightarrow}$
LT (Lines)	0.93*	0.93*	0.78*	0.79*	0.63*	0.69*
LT (Transitive)	0.51*	0.50*	0.49*	0.47*	0.51*	0.49*
DB1B	0.88*	0.88*	0.88*	0.88*	0.88*	0.88*
Wikispeedia	0.80*	0.42*	0.75*	0.33*	0.79*	0.41*
Wordmorph	0.70*	0.66*	0.60*	0.56*	0.69*	0.65*

	Weighted overlap τ_w					
	C_S^{\leftarrow}	C_S^{\rightarrow}	$C_{S'}^{\leftarrow}$	$C_{S'}^{\rightarrow}$	C_{SW}^{\leftarrow}	C_{SW}^{\rightarrow}
LT (Lines)	0.27	0.30	0.21	0.21	0.54	0.49
LT (Transitive)	0.47	0.48	0.45	0.45	0.45	0.46
DB1B	0.15	0.15	0.14	0.14	0.18	0.18
Wikispeedia	0.38	0.60	0.33	0.53	0.34	0.54
Wordmorph	0.57	0.57	0.45	0.47	0.45	0.47
	C_R^{\leftarrow}	C_R^{\rightarrow}	C_{RW}^{\leftarrow}	C_{RW}^{\rightarrow}	$C_{RW'}^{\leftarrow}$	$C_{RW'}^{\rightarrow}$
LT (Lines)	0.22	0.21	0.34	0.37	0.54	0.50
LT (Transitive)	0.45	0.45	0.46	0.46	0.45	0.46
DB1B	0.15	0.15	0.19	0.19	0.18	0.18
Wikispeedia	0.33	0.53	0.39	0.61	0.34	0.54
Wordmorph	0.43	0.45	0.57	0.57	0.44	0.46

Table 4.7 Span of ranking positions of the flow-based betweenness and closeness measures: For each node, its span of ranking positions (see also Section 4.3) was computed by subtracting its maximal ranking position with respect to any flow-based or variant or standard centrality from its minimal ranking position with respect to any flow-based variant or standard centrality. This was done separately for the betweenness and closeness variants. The table shows for each dataset the mean value of the spans of the rankings and the range of the ranking spans. The mean value is given in absolute ranking positions and, in parentheses, relative to the number of nodes.

Dataset	Betweenness		Closeness	
	Mean	Range	Mean	Range
LT (lines)	69.1 (25 %)	[2,197]	97.0 (36 %)	[12,221]
LT (transitive)	61.5 (23 %)	[0,218]	74.3 (28 %)	[6,234.5]
DB1B	62.1 (15 %)	[5,348]	57.0 (14 %)	[5,313.5]
Wikispeedia	845.2 (18 %)	[0,3743]	1988 (43 %)	[14,4458]
Wordmorph	288.1 (29 %)	[2,815]	444.8 (44 %)	[11,962]

4.6 Which nodes are impacted?

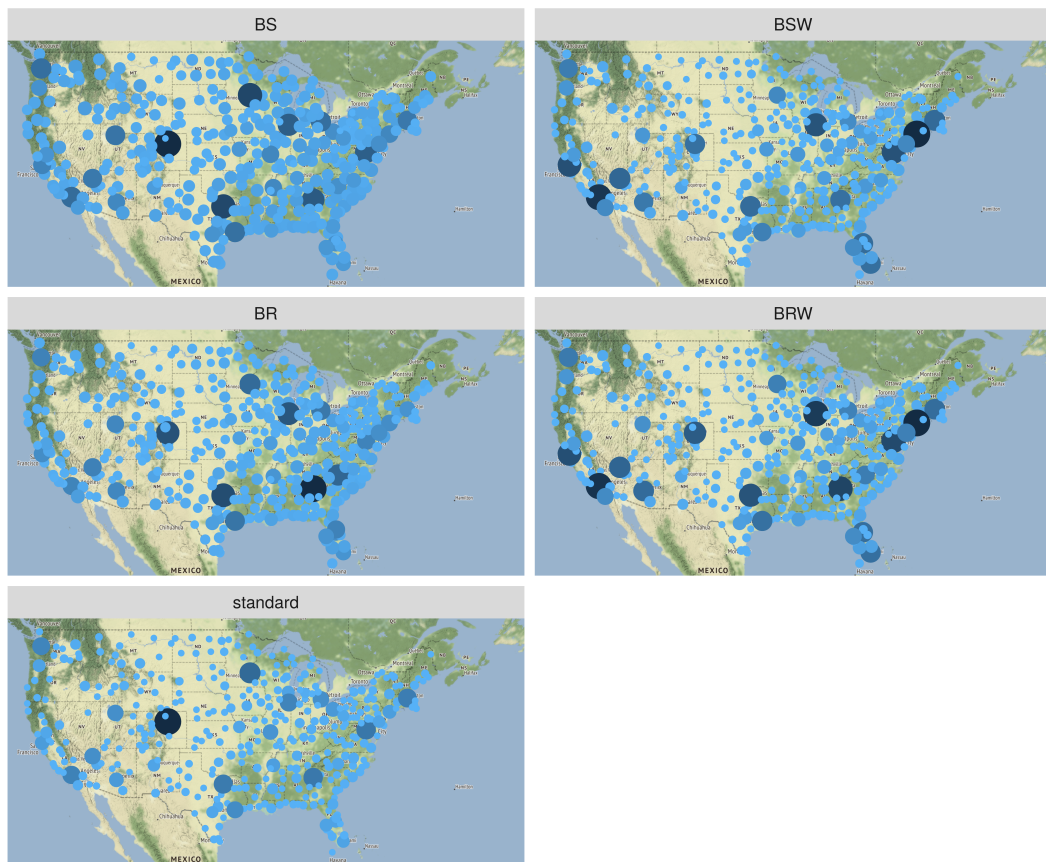
The last section compared the rankings from the flow-based centrality measures to the rankings of the corresponding standard centrality measure in order to check whether the standard centrality measures are robust against violations of the assumptions of their incorporated process model. This was done by comparing the complete rankings to each other. We found that, in general and for most flow-based centralities, the standard and flow-based centralities show a medium to high correlation. At the same time, there exist a non-negligible number of nodes whose ranking position changes considerably from one measure variant to another. This section provides material intended to explain these ranking variations. This is done in two steps: In Section 4.6.1, the analysis focuses on those nodes that are ranked high by at least one centrality variant. This approach is based on the fact that when analyzing a network with means of centrality measures, often only the most important nodes are of interest. We therefore consider the ranking behavior of those nodes that are central with respect to at least one measure variant. In the second step (see Section 4.6.2, the analysis focuses on the extreme cases, i.e., the rankings of those nodes are considered that show the largest ranking deviations among the centrality variants.

As illustration for the nodes' ranking behavior for the different centrality variants, the nodes and their centrality values are visualized on a map for those datasets containing nodes with geographic information. Figures 4.9 and 4.10 show the nodes' centrality values for betweenness variants for the air transportation dataset and for the London Transport dataset. Similarly, Figures 4.15, 4.16, and 4.17 show the visualization for the closeness variants.

4.6.1 Effect on high-ranked nodes

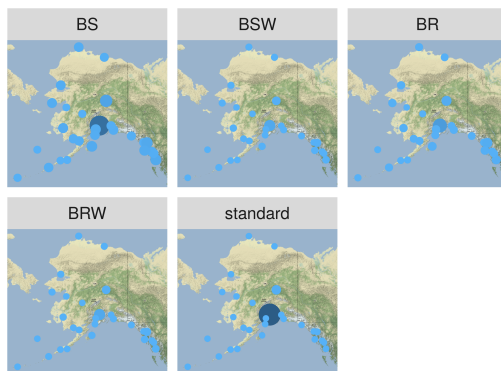
When applying a centrality measure to a network, the interest is mostly on the high-ranked nodes and less on the lower-ranked nodes. Especially for the topmost entries of the resulting ranking, high reliability of the results is desired. For this reason, in this section, we focus on those nodes that are among the highest-ranked entries for at least one centrality variant. If the flow-based and standard centralities approximately agree on the most important nodes while the changes in ranking positions rather occur in the lower part of the rankings, the effect is less relevant than if the flow-based and standard measures show large differences in their high-ranked nodes.

Figure 4.11 shows the ranking positions of those nodes that are among the ten highest-ranked nodes with respect to at least one flow-based betweenness centrality or with respect to standard centrality. Figure 4.14 shows the corresponding figures for the closeness variants. For each of these



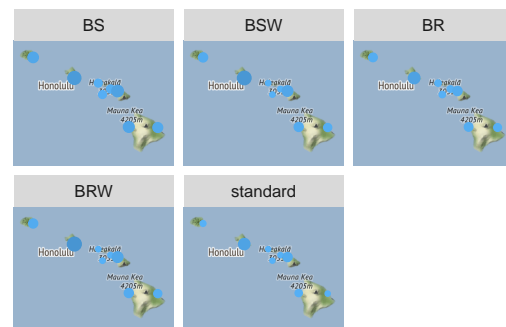
(a) US map.

Alaska



(b) Alaska.

Hawaii



(c) Hawaii.

Figure 4.9 Effect of flow-based betweenness measures on the nodes' centrality values for the DB1B dataset, visualized on a map: Each point represents a node (representing a city with airports) in its geographic location, color and size correspond to the centrality value of each variant.

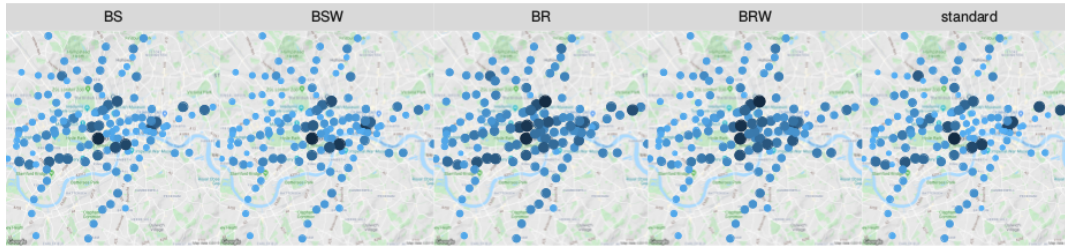


Figure 4.10 Effect of flow-based betweenness measures on the nodes' centrality values for the London Transport dataset (lines graph), visualized on a map: Each point represents a node (representing an London Underground Station) in its geographic location; color and size correspond to the centrality value of each variant.

nodes, the figures show its ranking position for each centrality variant.

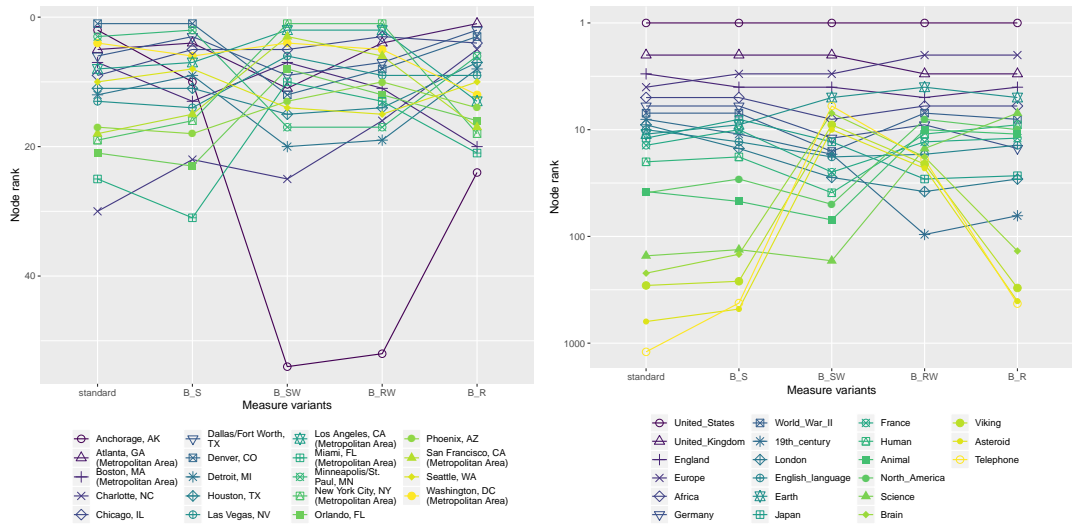
For all datasets and for both types of centrality measures, betweenness and closeness, we found that not the same set of ten nodes was ranked highest by all variants. There was considerable variation of rankings even among these nodes.

Betweenness measures

DB1B For the DB1B dataset, Figure 4.11a shows that all nodes ranked among the top ten by any measure variant, are at least among the top 60, most even among the top 30 nodes of all variants. Almost all nodes found among the top ten of at least one betweenness variant represent large⁶ airports. The ranking behavior of one node stands out: The node representing the airport of Anchorage in Alaska is ranked high (ranks 2 and 10, respectively) by standard betweenness centrality and variant B_S , but drops in importance with respect to the other variants. This effect can be explained when considering the structure of the network in more detail: The airport of Anchorage serves as a gateway between airports in the contiguous United States and the airports in the state of Alaska. It is not the case that any airport in Alaska can *only* be reached via Anchorage—there are direct flights from airports in the contiguous states to other airports in Alaska, but those are of almost no consequence for the computation of standard betweenness centrality, for the following reason: The network consists of a densely connected 46-core, a subgraph consisting of 56 nodes in which each node has a degree of at least 46. This subgraph almost forms a clique, therefore almost all nodes within this subgraph have a distance of 1 to each other, and all of them have a high betweenness value. All nodes of the dense subgraph represent airports located in the contiguous United States; none of them represents an airport in Alaska. The reason why the node Anchorage is ranked high by standard betweenness centrality is that Anchorage is well connected to nodes in the dense subgraph and to all Alaskan airports. Figure 4.12a shows the relevant extract of the network, i.e., all Alaskan airports and their direct neighbors. The nodes are colored depending on the k -core of the network they belong to. Red nodes belong to the large dense subgraph, i.e., the 46-core, while Anchorage and Honolulu (colored in orange) belong to the 44-core. There are a few Alaskan airports belonging to a 15- or 10-core, but most Alaskan airports are only connected to one or two other airports. Anchorage, however, is well-connected to the densely connected subgraph—it is directly connected to 24 of those nodes—and more than half of Alaskan airports can only be reached via Anchorage. This is why Anchorage is ranked high by standard betweenness centrality—even higher than almost all nodes within the dense subgraph. When the observed number of passengers traveling to an airport is incorporated into betweenness centrality, as done by the variants B_{SW} and B_{RW} , the relative importance of the node Anchorage decreases. This is because only 0.5% of all journeys of the dataset have an airport in Alaska as their destination.

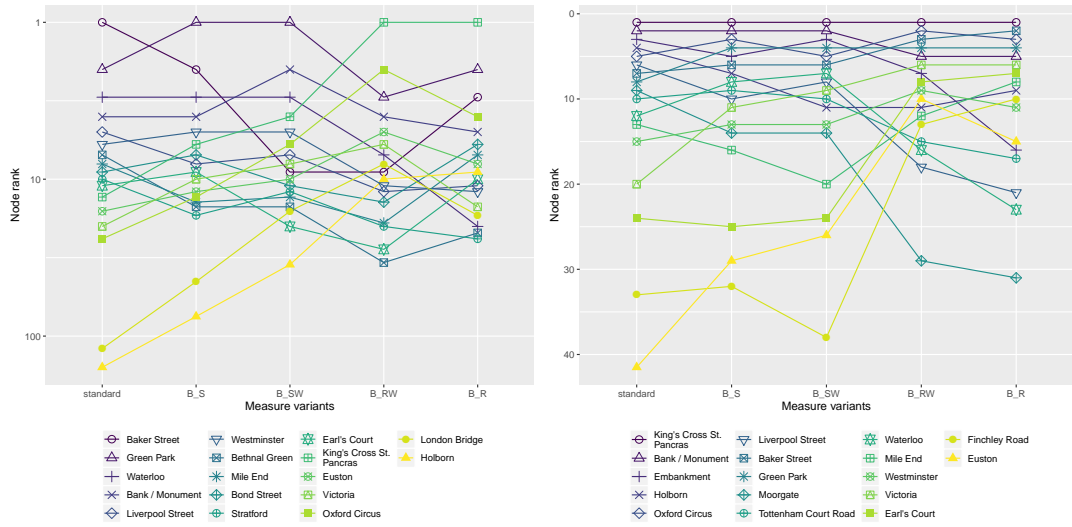
⁶as classified by the Federal Aviation Administration (FAA), part of the United States Department of Transportation[Adm17]

4.6 Which nodes are impacted?

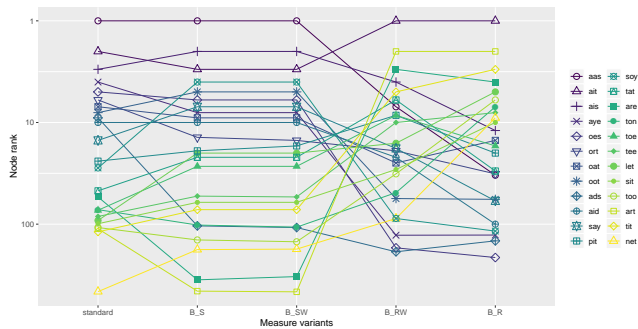


(a) DB1B.

(b) Wikipedia. Note logarithmic scale on y -axis.

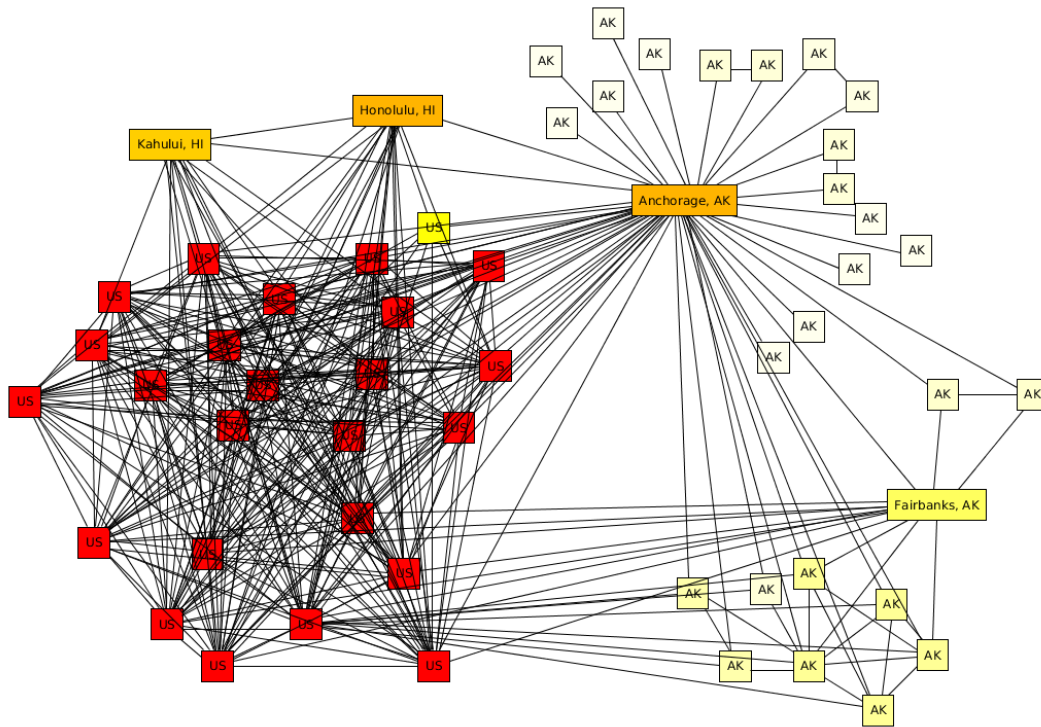


(c) London Transport (lines graph). Note logarithmic scale on y -axis. (d) London Transport (transitive).

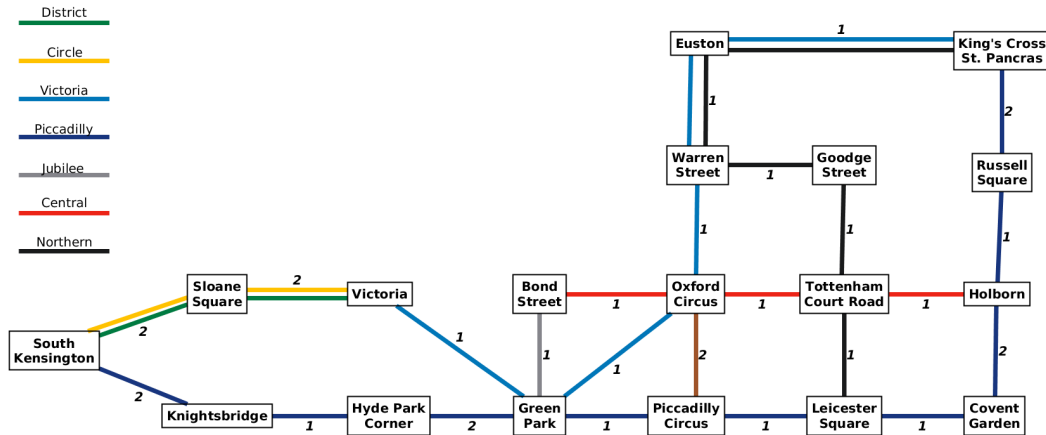


(e) Wordmorph.

Figure 4.11 Top 10 nodes (betweenness) Ranking positions with respect to all betweenness centrality variants of those nodes which are among the 10 most central nodes with respect to at least one centrality variant. Top nodes have rank 1 (top of each plot), the order of the line colors is due to the node's standard betweenness centrality.



(a) An extract of the DB1B air transportation network: All Alaskan airports and their direct neighbors. The color shows to which k -core the nodes belong in the complete network: Red nodes belong to the largest and most densely connected core of the network. Nodes representing airports in Alaska are labeled by AK, nodes representing airports in contiguous United States are labeled by US.



(b) A small extract of the lines graph for the London transportation system. The edge labels indicate the travel time between the stations which are used as edge weights; the edge colors indicate which Underground lines serve the corresponding connection.

Figure 4.12 Extracts of the air transportation network of the DB1B dataset and the lines network of London Transport dataset.

London Transport For the dataset containing public transport journeys within London, we found that the results are different for the two network versions (see Figure 4.11c and 4.11d): While in the transitive network version, the node representing the station King’s Cross St. Pancras is the most central node with respect to all flow-based betweenness variants and to standard betweenness centrality, for the lines graph variant, this node is the most central one only with respect to the variants B_R and B_{RW} . For the other variants, this node is also among the top ten nodes, but less highly ranked. For the lines graph variant, it is the node representing the station Baker Street that is ranked as the most central node by standard betweenness centrality. Both stations, King’s Cross St. Pancras and Baker Street, serve as junction points in the London Underground network. King’s Cross St. Pancras is the only station in the network serving six different Underground lines, while the Baker Street station serves five different Underground lines. Therefore, and due to the structure of the network (see Figure 4.13)—a more densely connected subgraph in the “center” and chain-like structures appended to it—, both nodes are contained in many shortest paths. The Baker Street station, however, is slightly better positioned in the network in the sense that it is contained in more shortest paths than King’s Cross St. Pancras. When taking into account the real usage of the system, the top ranking of the node Baker Street is lost: In reality, there are more passengers traveling through King’s Cross St. Pancras than through Baker Street.

There is an effect observable here that is due to the modeling of the system: For the lines network variant, the nodes Holborn and London Bridge are not among the top 100 nodes for standard betweenness centrality. But they are among the top 100 nodes for the variants B_S and B_{SW} , and even among the top ten for the variants B_R and B_{RW} . When we investigated the reason for this rise of importance, it turned out that this is caused by the modeling of the system. The system is modeled as *one* network where there exists an edge from one node to another if there is a non-stop Underground connection from the one station to the other. The edges are weighted by travel time; however, additional time needed for changing lines (getting off a train, changing platforms, and getting onto another train) is not taken into account. An algorithm computing the shortest path on this network may therefore yield paths that—although they are the shortest in the graph representation and taking into account the edge weights—are not the fastest (or most convenient) connection in reality. In reality—depending on the actual timetables—, it is often advantageous (and more convenient) to include fewer line changes in one’s travel plan. A shortest path algorithm, however, may—in extreme cases— yield paths where each edge corresponds to a different line and a change of lines would be necessary at each intermediate node. The ranking behavior of the nodes Holborn and London Bridge is caused by this effect. Consider Figure 4.12b which shows an extract of the lines network where edges are drawn several times if a connection is served by several lines. In most cases, passengers traveling from King’s Cross St. Pancras to South Kensington, for example, will take the Piccadilly line which takes 13 minutes and does not require any change of lines. The shortest path in the network, however, is: King’s Cross St. Pancras to Green Park via the Victoria line, and then to South Kensington via the Piccadilly line, which yields a path length of 9 minutes in total. Since for many node pairs, the shortest path between them is going through this subgraph, this effect occurs for the computation of the shortest path between all these node pairs. Therefore, the node is not ranked high by standard betweenness centrality (because it contains an unrealistic shortest path model), is ranked a bit higher by the variants B_S and B_{SW} (because in these variants, many node pairs on opposite ends of the network are not included in the computation), and is ranked among the top ten for the variants B_R and B_{RW} (because in these measures, journeys actually taken are considered which often do contain the node Holborn). The effect is also visible for other nodes in this network extract besides Holborn, however, these are not among the top ten of any measure and therefore not contained in Figure 4.11c.

The same effect occurs in the transitive variant of the network. Here, we observe the opposite effect: Nodes *lose* importance when real passenger journeys are taken into account as in the case of the node Moorgate. Moorgate is among the top ten nodes for the variants counting shortest paths, i.e., standard betweenness centrality, and the variants B_S and B_{SW} , but is only ranked in position 31, respectively 29, for the variants B_R and B_{RW} . For real passenger journeys, there is often no need

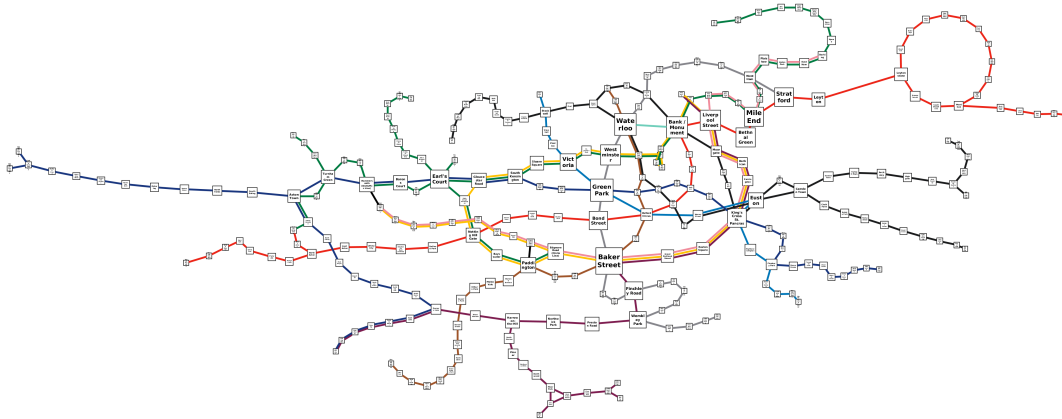


Figure 4.13 London Transport lines graph: The edges are colored according to which Underground line serves this connection (and drawn several times if more than one line serves a connection); the size of the nodes corresponds to its standard betweenness value. The edge weights are not shown.

to change trains at Moorgate which is why (in the transitive network representation), this node is not contained in the passenger journeys and therefore ranked less high for B_R and B_{RW} . In the graph representation, however, there exists a shorter path that does contain Moorgate, which results in a higher ranking position for the variants counting shortest paths.

Therefore, in future work, it might worth considering—depending on the research question—modeling the system in a different way such that the time needed for changing lines is included in the network.

Wikipedia For the dataset containing game logs of the game Wikipedia, we can make several observations (see Figure 4.11b). Among the nodes contained in Figure 4.11b, i.e., nodes that are among the top ten of at least one betweenness measure variant, the differences in ranking positions are higher than for all other datasets. There is a node that is among the top ten for one measure, and not even among the top 1000 for other measures. At the same time, for all measure variants, the top five nodes are rather stable in their ranking position. The node representing the article on the UNITED STATES is the most central node for all measure variants which reflects a popular strategy when playing the game Wikipedia: Players often⁷ use this article as a landmark in their navigation. They first navigate to the article on the UNITED STATES (which is long and has a large number of outgoing links⁸) and from there to the target article. In Figure 4.11b, it can also be observed that there are four nodes (representing the articles on BRAIN, VIKING, ASTEROID, and TELEPHONE) with considerably higher values in the weighted betweenness variants, i.e., B_{SW} and B_{RW} . This effect is due to the data collection method described by West and Leskovec [WL12b]: Normally, when a player starts a game, they can either choose a start and target article, or two articles are drawn uniformly at random and suggested as the start and target article. However, for a specific research question, for a limited time period, there were four node pairs that were suggested to the players more often. This is why the node pairs (ASTEROID, VIKING), (BRAIN, TELEPHONE), (THEATRE, ZEBRA), and (PYRAMID, BEAN) occur with a higher frequency in the dataset (in descending order). Therefore, the nodes ASTEROID, VIKING, BRAIN, and TELEPHONE gain in importance when the observed amount of flow between node pairs is incorporated into the measure variants, i.e., with B_{SW} and B_{RW} . For the four other nodes (THEATRE, ZEBRA, PYRAMID, and BEAN), this effect is also observable, but it is less strong, so they do not reach the top ten nodes of the ranking of any measure and are therefore not contained in the figure.

⁷the article on the UNITED STATES is contained in approximately 16% of all game logs

⁸The article on the UNITED STATES actually has the largest number of incoming links and the largest number of outgoing links of all nodes of the network.

Wordmorph For the other dataset containing game logs (see Figure 4.11e), a high similarity of the rankings by the measures B_S and B_{SW} , and of the rankings by B_R and B_{RW} can be found. This is plausible because of the source-target-pairs used, 99% were the source and target of exactly one game log, and the remaining of at most three game logs. Therefore, the weighted and unweighted variants show similar ranking results. However, when comparing the variants counting shortest paths to those counting real trajectories, differences in the rankings can be observed. The nodes ART and ARE increase their ranking position from 111 to 2 and from 43 to 3, respectively. At the same time, the nodes OES, AYE, and SOY lose ranking positions when we switch from counting shortest paths to counting real trajectories. The node AIT stays among the top three nodes for every measure. It is noticeable that especially those nodes gain in importance that are more common in daily language (such as ARE, ART, SIT, etc) while unusual words—possibly unknown to players—(such as AAS, OES, or ORT) have a good position in the network due to their betweenness value, but are not included in many game logs.

Closeness measures

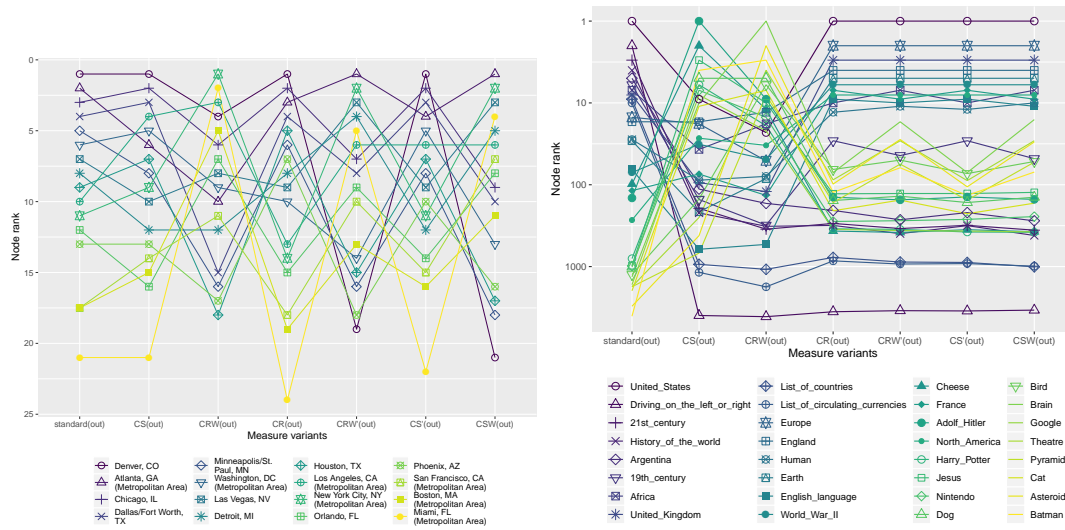
Figure 4.14 shows the ranking behavior of the high-ranked nodes for each closeness centrality variant, i.e., as before for the betweenness variants, the ranking positions of all nodes that are among the ten highest-ranked nodes with respect to at least one measure variant. This was done separately for the in- and out-variants. We only show and describe the results for the out-variants; the figures for the in-variants can be found in the Appendix.

DB1B For the dataset containing air transportation journeys (see Figure 4.14a, it can be seen that there is only a small variation of ranking positions among the considered nodes. Particularly the rankings of the measures C_S , $C_{S'}$, and C_R rank the nodes in a similar way. When weights proportional to the amount of observed flow between node pairs are incorporated into the measures, there are considerable⁹ changes in the rankings. When considering the measure variants C_{SW} and $C_{RW'}$, it can be observed that there is a small set of nodes that seem to switch ranking positions: The nodes representing the airports Denver, Minneapolis/St. Paul, and Phoenix are among the most central nodes with respect to all other variants, but decrease in importance for C_{SW} and $C_{RW'}$, while other nodes such as Miami increase their ranking position by more than twenty positions. This effect is due to the high amount of passenger traffic between the airports Miami and New York. Between these two airports, the second highest amount of traffic is observed in the dataset (more passenger journeys in the dataset occurred only between San Francisco and Los Angeles). Furthermore the network consists of a densely connected subgraph, a 46-core in which all nodes shown in Figure 4.14a are contained; therefore, all nodes within this subgraph have a distance of 1 to each other. Particularly those nodes shown in Figure 4.14a have a distance of at most 3 to any other node (and a distance of 1 or 2 to most other nodes). Thus, when the distance between two nodes is weighted proportionally to the amount of traffic between them, a node such as Miami with a highly demanded connection to another well-connected node can boost its centrality ranking and get ranked higher than nodes whose unweighted average distance to all other nodes is smaller.

London Transport Figure 4.14c and 4.14d show the results of the closeness variants for the two network variants of the London Transport dataset. As before for the betweenness measure variants, we found that the results differ notably for the two network variants. While for the transitive network version, all shown nodes are among the top 60 of each measure, for the lines network variants, there are nodes that are among the top ten for one measure and not even among the top 100 for other measures. Furthermore, for the transitive network variant, there are only small variations in ranking positions among the flow-based centrality variants: When comparing standard closeness to the flow-based closeness measures, the top ten nodes are almost disjoint; when comparing the flow-based measures among each other, the nodes' ranking positions are rather stable. This is different for the lines network variant. Here, especially the variants $C_{RW'}$ and C_{SW} show

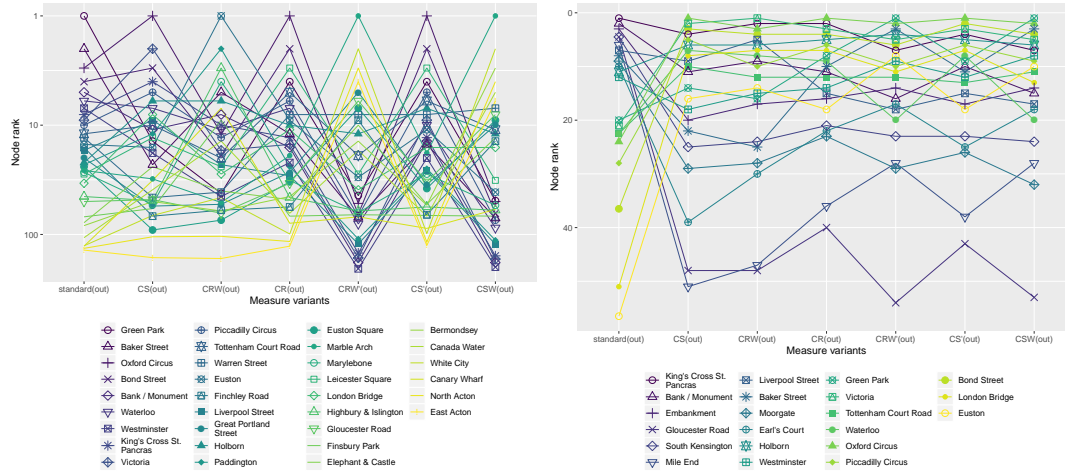
⁹In absolute numbers, the changes observed here are still small: All considered nodes are still among the top 25 nodes.

4 Flow-based centrality measures



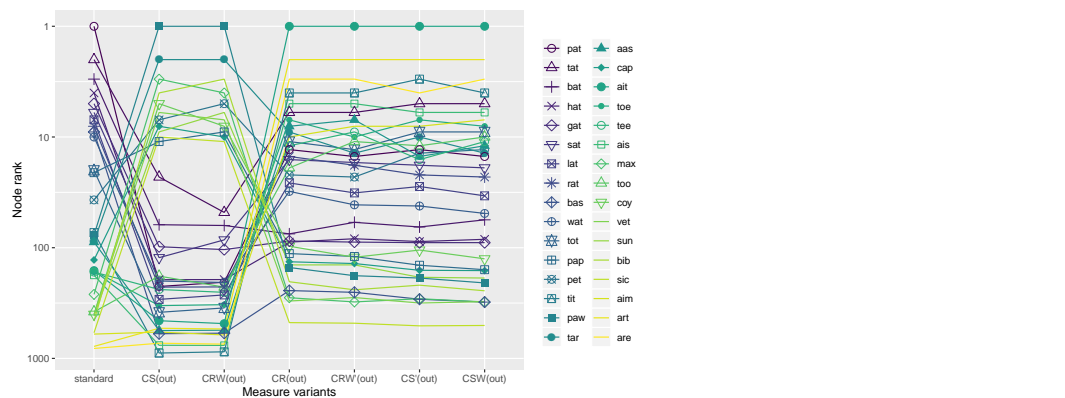
(a) DB1B.

(b) Wikipedia. Note logarithmic scale on y-axis.



(c) London Transport (lines graph). Note logarithmic scale on y-axis.

(d) London Transport (transitive).



(e) Wordmorph. Note logarithmic scale on y-axis.

Figure 4.14 Top ten nodes (closeness) Ranking positions with respect to all (out-)closeness centrality variants of those nodes that are among the ten most central nodes with respect to at least one centrality variant. Top nodes have rank 1 (top of each plot), the order of the line colors is due to the node's standard closeness centrality.

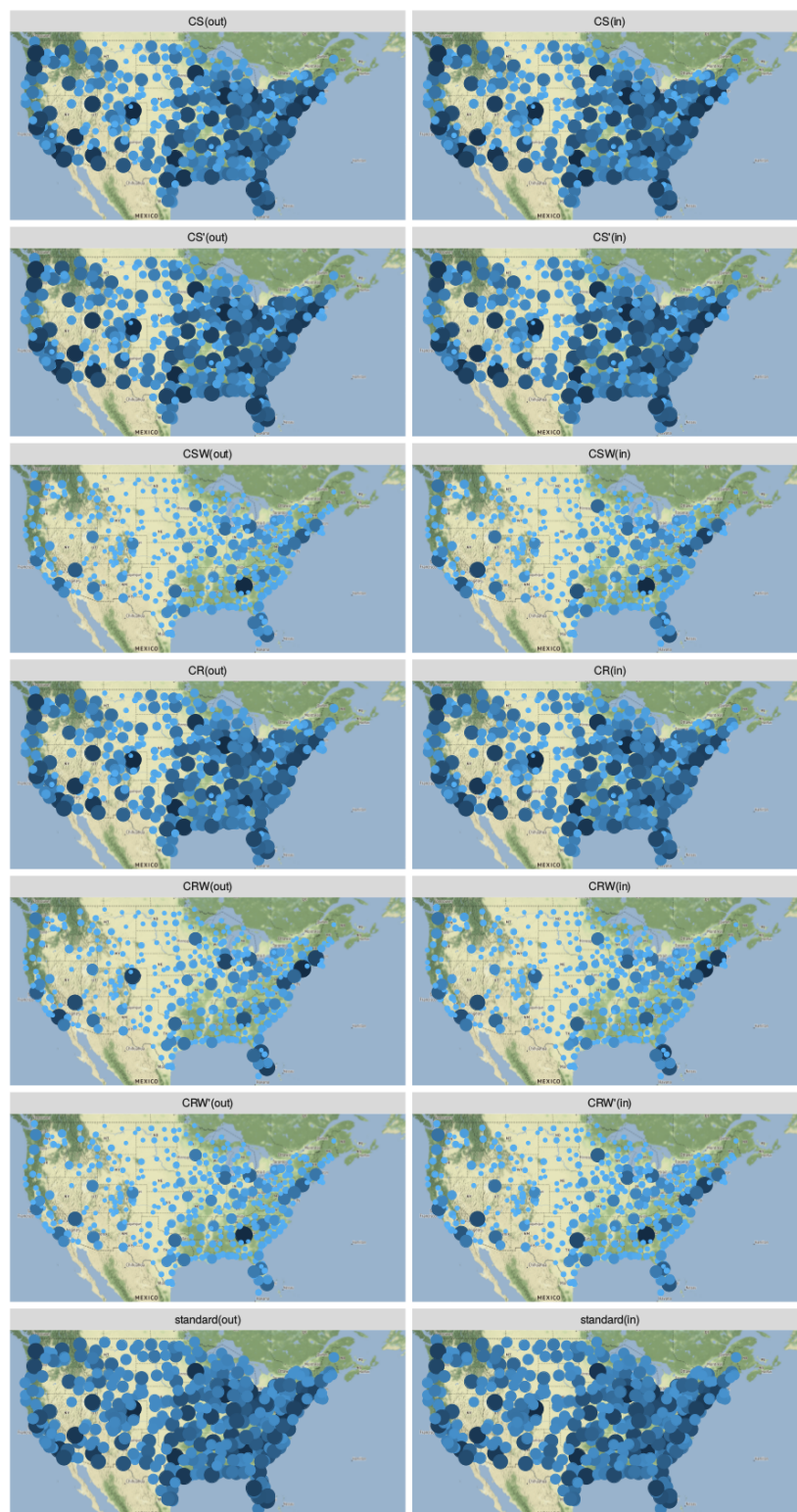
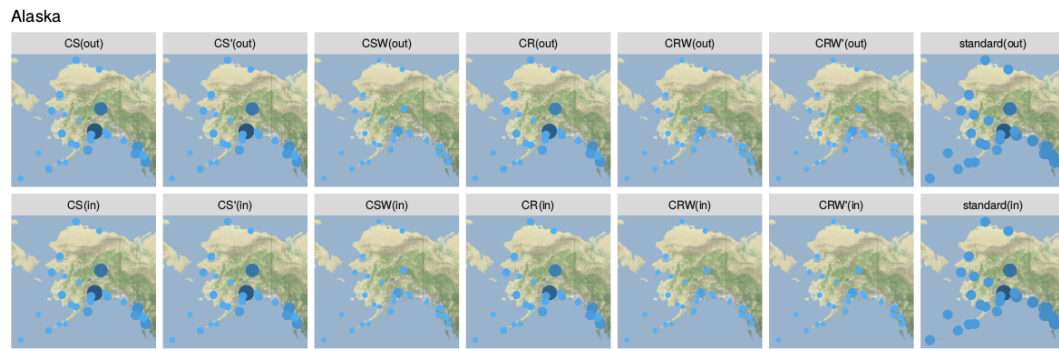
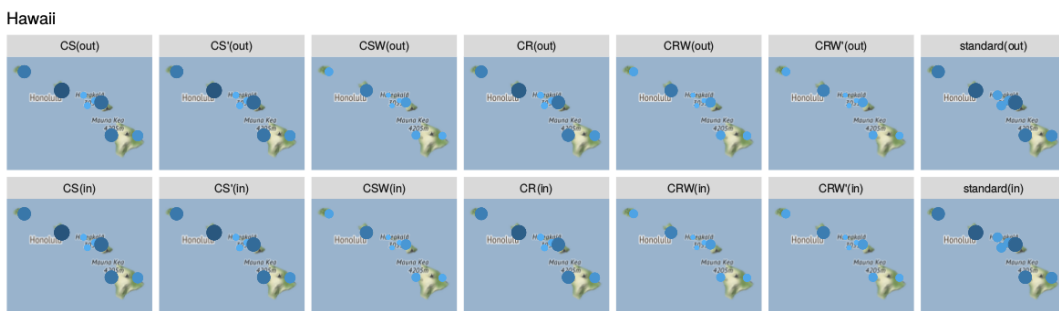


Figure 4.15 Closeness values of the flow-based variants for the air transportation dataset (DB1B) visualized on a US map (for Alaska and Hawaii, see Figure 4.16). Each node representing a city area is shown by a point in its geographic location; size and color correspond to the centrality value of each variant.



(a) Alaska.



(b) Hawaii.

Figure 4.16 Values of flow-based closeness variants visualized on a map, as in Figure 4.15; here for Alaska and Hawaii.

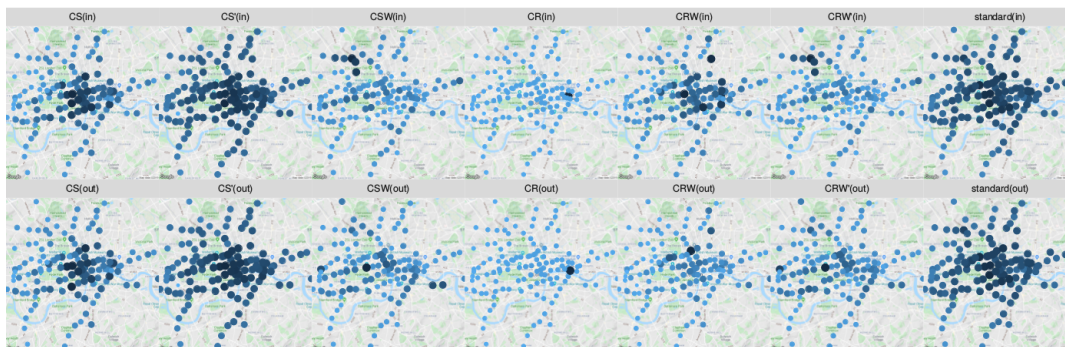


Figure 4.17 Values of flow-based closeness variants on a map of London (London Transport dataset (lines graph)). Each node representing a London Underground station is shown as a point in its geographic location; size and color correspond to the value of the corresponding centrality variant.

a substantially different set of nodes rated as important.

Wikispeedia For this dataset, incorporating observed trajectories into closeness centrality had a much larger effect than for any other dataset (see Figure 4.14b): Even among the highly ranked nodes contained in the figure, there are nodes whose ranking position increases or decreases by more than 1000 positions. Interestingly, there appear to be three types of behavior: the ranking behavior of the standard closeness centrality, the ranking behavior of the measure variants C_S and C_{RW} which are rather similar, and the ranking behavior of the remaining measure variants which is almost identical. For the measure variant C_{RW} (and, less strongly, also for C_S), we again see the impact of the data collection method: As mentioned in Section 4.6.1, for a specific research question, West and Leskovec modified their game platform in the sense that four node pairs were suggested to the players with increased frequency [WL12b]. The impact of the increased source-target frequency of those node pairs can also be observed for the closeness variants: The ranking positions of the corresponding source nodes (ASTEROID, BRAIN, THEATRE, and PYRAMID) increased by more than 1000 positions and reached the top eight nodes of this measure variant. Since players were also allowed to choose a source and target node themselves, several effects of this can be observed in the ranking behavior: For some reason, the article BATMAN was a popular start article (5th most frequently used source article, behind the four promoted ones) which is reflected by its increased ranking position with respect to the measure variant C_{RW} : Here, it is in position 2 (while being in rank 4044 for standard out-closeness centrality). When considering the rankings of C_S , the articles on JESUS and ADOLF HITLER are among the top three nodes. We speculate that this is due to two popular game variants—5 clicks to Jesus and Clicks to Hitler which are believed to be the original forms of the game. In these variants, a player needs to navigate from a randomly chosen article to the article on JESUS or ADOLF HITLER, respectively. We can only speculate that due to the popularity of these variants, the articles on JESUS and ADOLF HITLER were popular source and target nodes. The effect can be seen in the increase of their ranking in the variant C_S . The effect is stronger for the in-variants where these two articles jumped to ranking position 1 and 2 for C_S and C_{RW} , respectively (see Figure A.5b in the Appendix).

An interesting effect can be observed for the articles DRIVING ON THE LEFT OR RIGHT, LIST OF COUNTRIES, and LIST OF CIRCULATING CURRENCIES. All three articles contain a complete list of all countries with links to the corresponding articles which results in a high ranking of standard out-closeness centrality. However, since these articles were used not at all or only very few times by the players, their ranking position is constantly low for all flow-based closeness variants.

Wordmorph For the closeness variants, there are similar observations for both game datasets, Wikispeedia and Wordmorph. The ranking variation among the considered nodes—top ten with respect to at least one measure variant—is larger for these datasets than for the transportation data sets. For Wordmorph, we found nodes with a span of ranking positions of more than 700 positions. The node representing the word ARE for example increased its ranking position from 813 for standard closeness to position 3 or 4 for several flow-based measure variants. Note furthermore that the rankings with respect to the variants C_R , C_{RW} , C_S , and C_{SW} were rather stable for the considered nodes.

4.6.2 Which nodes are impacted most?

The previous section focused on the impact of high-ranked nodes, i.e., nodes that are ranked high by at least one centrality variant. We found that even among those nodes, there are large variations in ranking positions among the flow-based centrality variants. In a second step, we considered the extreme cases, i.e., those nodes that are affected most by incorporating the properties of the real-world network flow and those that are affected least. In order to see the impact on the nodes, a node's highest ranking position among all centrality variants (flow-based and standard) and its lowest ranking position were considered. This was done separately for the betweenness

and closeness variants. Figure 4.18 shows the nodes' highest and lowest ranking positions for the betweenness variants. The difference between minimal and maximal ranking position was introduced as the *span* of a node in Section 4.3. Tables 4.8 to 4.12 list for each dataset the most stable nodes, i.e., the nodes with the smallest *span*, and the most unstable nodes, i.e., the nodes with the largest *span*. Although there were already considerable changes in the ranking positions among the highly ranked nodes (see previous section), Figure 4.18 shows that there was also an impact on lower-ranked nodes.

In the following, we will only discuss the results concerning the betweenness variants; the corresponding figures for the closeness variants can be found in the Appendix.

DB1B As already shown in Table 4.7, it is also visually obvious that the impact was the smallest for the DB1B dataset. For this dataset, the majority of nodes is plotted close to the identity line, meaning that their highest and lowest ranking positions are close to each other. There are, however, nodes with considerable differences in their ranking positions: Table 4.8 shows the nodes with the largest span of ranking positions. The airports Atlantic City (New Jersey), Stockton (California), and Rockford (Illinois) have the largest span of ranking positions, more than 300 positions for each of them. The table also shows the nodes' ranking positions with respect to each centrality variant: For the four most unstable nodes, their instability is mainly due to their high ranking position of B_R and their low ranking position of B_S . When considering which nodes are the most stable and the most unstable, it can be observed that nodes representing large cities seem to be stable, while nodes representing smaller cities are among the unstable nodes. We quantified this speculation by using data provided by the Federal Aviation Administration, part of the United States Department of Transportation. They provide a categorization of US airports by purpose and by their number of annual passenger boardings. They categorize airports into the following categories [Adm17]:

- Commercial service airports: publicly owned airports with scheduled passenger service and at least 2500 passenger boardings per year;
- cargo service airports: airports served by cargo aircrafts, might also be a commercial service airport;
- reliever airports: airports to relieve congestion at commercial service airports; and
- general aviation airports: airports without scheduled passenger service or with less than 2500 passenger boardings per year.

Commercial service airports are further categorized by the percentage of their annual passenger boardings:

- *Primary large hubs*: more than 1% of all annual passenger boardings;
- *primary medium hubs*: between 0.25% and 1% of passenger boardings;
- *primary small hubs*: between 0.05% and 0.25% of all passenger boardings;
- *primary non-hubs*: at most 0.05% of all passenger boardings, but at least 10000 passenger boardings; and
- *non-primary airports*: between 2500 and 10000 passenger boardings per year.

For the DB1B dataset at hand, we assigned a category to each node according to this categorization: If a node represents a city containing more than one airport, the category of its largest airport was assigned. The dataset then contained 23 primary large hubs, 27 primary medium hubs, 66 primary small hubs, 228 primary non-hubs, 52 cargo service airports, and 19 general aviation airports. Figure 4.19 shows the nodes' span of ranking positions according to their categorization. It can be seen that especially nodes representing cities with smaller airports, i.e., labeled as primary non-hubs and non-primary airports, show large spans of ranking positions, significantly higher than for large and medium hubs. The ranking positions of these, large and medium hubs, are rather stable. Their span of ranking positions is only 11 and 17 positions on average (median) and shows very small variance. Small hubs have a median ranking span of 29 positions and larger variance. However, the nodes la-

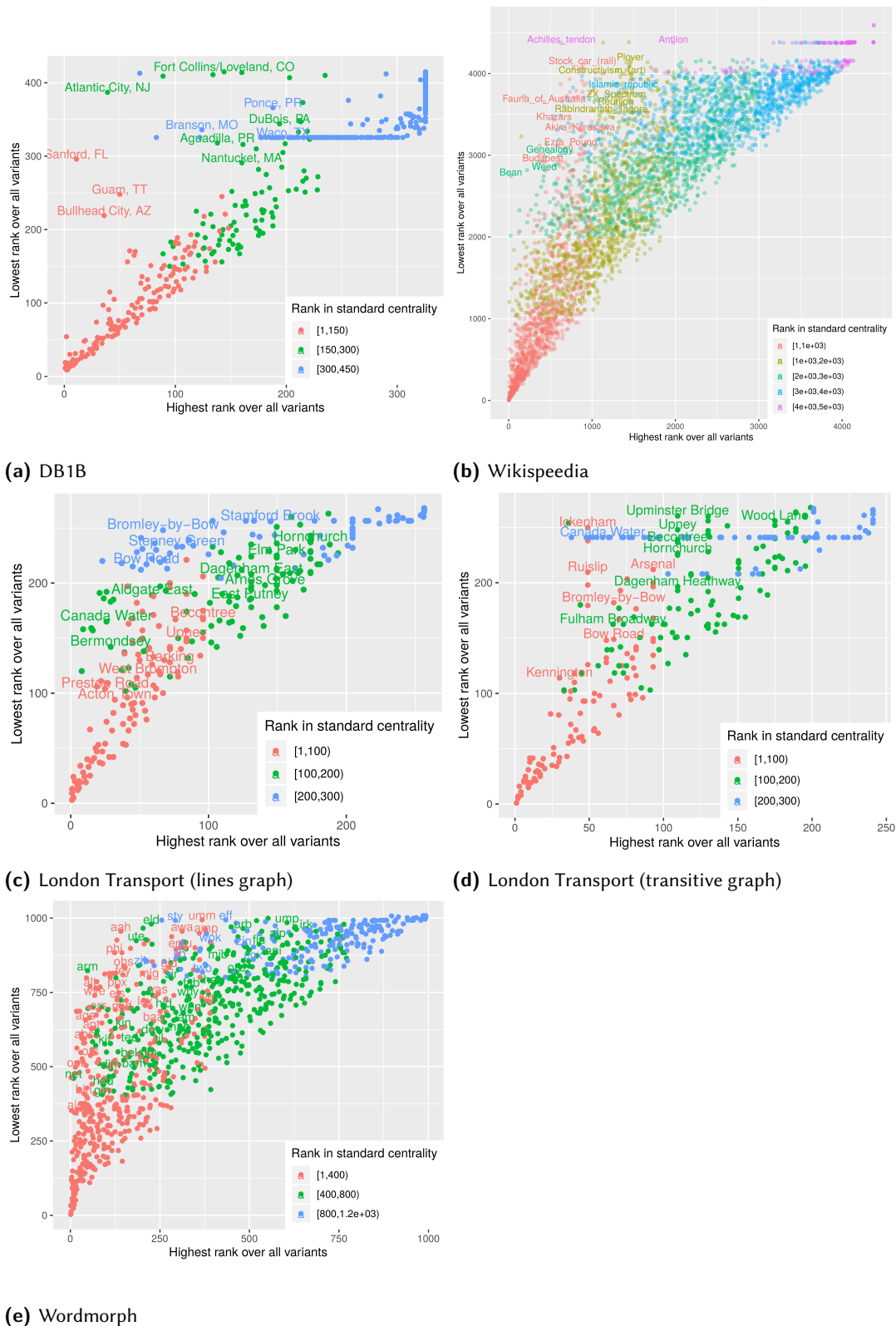


Figure 4.18 Min-max-plot for rankings for flow-based betweenness centrality variants: Each node is represented by a point and its highest ranking position over all betweenness variants (flow-based and standard betweenness centrality) is plotted against its lowest ranking position. The color of the points indicates the nodes' ranking with respect to standard betweenness centrality.

Table 4.8 The most stable nodes with respect to their ranking position among the betweenness variants (standard and flow-based variants) and the most unstable nodes, for the DB1B dataset. $span(v)$ describes the nodes' span of rankings, i.e., the difference between its highest and its lowest ranking position among all centrality variants. In total, the network contains 415 nodes.

DB1B: Most stable nodes						
Node	standard	B_S	B_{SW}	B_{RW}	B_R	$span(v)$
Kansas City, MO	29	28	26	26	31	5
Chicago, IL	9	5	5	3	4	6
Dallas/Fort Worth, TX	6	3	9	7	2	7
Seattle, WA	10	8	14	15	10	7
Colorado Springs, CO	73	67	72	72	65	8
Washington, DC	4	6	4	5	12	8
Houston, TX	11	11	15	14	7	8
Las Vegas, NV	13	14	6	9	9	8
Phoenix, AZ	17	18	13	10	14	8
Tampa, FL	22	27	19	21	22	8
DB1B: Most unstable nodes						
Node	standard	B_S	B_{SW}	B_{RW}	B_R	$span(v)$
Atlantic City, NJ	162	387	88	87	39	348
Stockton, CA	325.5	413	223	223	68	345
Rockford, IL	177	409	207	207	89	320
Sanford, FL	61	296	101	103	11	285
Punta Gorda, FL	224	411	235	235	134	277
Fort Collins/Loveland, CO	234	415	245	245	144	271
Youngstown/Warren, OH	231	414	267	267	160	254
Hilo, HI	325.5	198	85	83	234	242.5
Branson, MO	325.5	336	195	194	124	212
Owensboro, KY	206	407	296	296	203	204

beled as non-hubs and as non-primary airports exhibit a ranking span of 63.25 and 60, respectively, on average (median) and show the largest variance. Therefore, for this dataset, nodes representing cities with large and medium airports are embedded in the network such that their network position and their importance with respect to the actual network flow match well: Incorporating real-world network flow into the betweenness measure impacts the nodes' ranking position only in a minor way. This is different for nodes representing cities with smaller airports, particularly those labeled as non-hubs and as non-primary airports. Here, measuring their importance using standard betweenness centrality or any of the flow-based variants can lead to considerably different results.

London Transport For the London Transport dataset, we observed that the results were different for the two network representations of the system. Figures 4.18c and 4.18d show that the rank variation is generally larger for the transitive graph than for the lines graph. Tables 4.9 and 4.10 reveal that there are mostly different nodes that are stable and unstable for the two network representations. In order to learn which type of nodes are affected most (or least) by switching between the different betweenness variants, we used external data about properties of the stations. First, we compared the average span of ranking positions with the position of the corresponding station within the area of London. For this reason, we used the station's information about the public transport zone to which it belongs: As for most urban public transport systems, the stations are grouped into zones. Stations in the inner city belong to zone 1, while zones 2 to 9 are concentric around it. Second, we used another attribute provided by Transport for London [Tra17], the number of daily passengers exiting a station (from TfL Rolling Origin and Destination Survey 2017). The

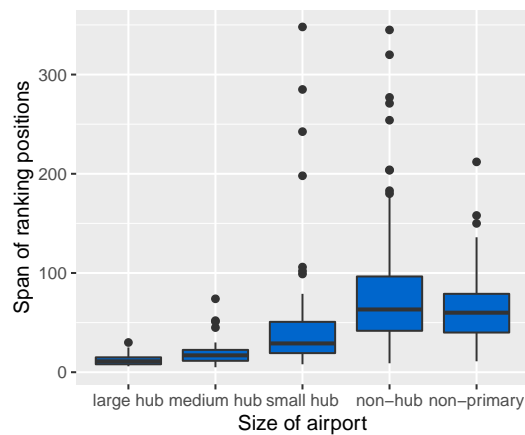


Figure 4.19 For the DB1B dataset, there exists an official classification of airports based on the airport’s purpose and its annual passenger numbers. The node’s span of ranking positions for the betweenness variants is shown against their official classification.

number of passengers exiting a station is assigned to four equally sized intervals which we used as a station attribute. Figure 4.20 shows the relationship between these two attributes and the average span of ranking positions. Unlike before for the DB1B dataset where a clear connection between the impact on the ranking position and the size of the airport could be observed, this is not the case here. Neither for the geographic location of the stations expressed by their zone, nor for the actual passenger traffic, can a clear connection to the span of ranking positions be observed. Therefore, the impact of incorporating real-world network flow into betweenness centrality on the resulting nodes’ ranking seems to be uncorrelated for both network representations and both considered station attributes.

Wikipedia Figure 4.18b supports the finding of the previous sections that for the Wikipedia dataset, there was an effect on the nodes’ rankings when the properties of the real-world network flow were incorporated into the centrality measures, here betweenness. Figure 4.18b reveals that there are nodes that are among the highest-ranked nodes for one measure variant and among the least highly ranked nodes for another measure. There is, for example, the node representing the article on *ACHILLES TENDON* which is among the least central nodes for standard betweenness and the unweighted flow-based variants, but reaches the ranking position 632, respectively 917.5 for B_{SW} and B_{RW} . For the stable nodes, on the other hand, we found that there are two nodes that have almost no changes in ranking position among all variants: The node *UNITED STATES* is the most central node for all variants, and the node *UNITED KINGDOM* ranks in position 2 or 3 for all variants. Apart from many nodes (more than 300) that are all in the same low-ranking position and therefore show a small span of ranking positions (see Table 4.11), there are only very few nodes with such a small span of ranking positions. It can be observed that especially articles on countries are stable nodes.

Wordmorph For the second game dataset, i.e., Wordmorph, Figure 4.18e confirms that a considerable number of nodes were rated very differently by the different betweenness centrality variants. Several nodes lose or gain more than 700 ranking positions when changing from one variant to another. Table 4.12 lists the most stable and most unstable nodes for this dataset, i.e., those nodes with the smallest and largest value of $span(v)$. Like for the other datasets, it can be hypothesized that the large span of ranking positions could be explained by an external node attribute. For this dataset, a plausible hypothesis is that words that are used more often in natural language and in texts will be more stable in their ranking behavior. Therefore, we used a frequency list of English words based on the Corpus of Contemporary American English provided by [Dav8]. Based on a

Table 4.9 The most stable nodes with respect to their ranking position among the betweenness variants (standard and flow-based variants) and the most unstable nodes, for the London Transport dataset (lines graph). $span(v)$ describes the nodes' span of rankings, i.e., the difference between its highest and its lowest ranking position among all centrality variants. In total, the network contains 268 nodes.

London Transport (lines graph): Most stable nodes						
Node	B_c	B_S	B_{SW}	B_{RW}	B_R	$span(v)$
Green Park	2	1	1	3	2	2
Bank / Monument	4	4	2	4	5	3
Upminster	256.5	259	257	256	260	4
Watford	256.5	260	261	258	256	5
Cockfosters	256.5	255	262	259	262	7
Liverpool Street	5	8	7	12	11	7
Westminster	6	5	5	11	12	7
Harrow & Wealdstone	256.5	261.5	254	254	257	7.5
London Transport (lines graph): Most unstable nodes						
Leicester Square	220	149	79	23	31	197
Swiss Cottage	241	226	212	87	51	190
Bromley-by-Bow	242	248	238	98	67	181
Charing Cross	218	147	93	37	84	181
St. John's Wood	228	205	189	79	48	180
Blackfriars	231	197	139	52	80	179
Covent Garden	236	194	142	61	64	175
North Greenwich	215	152	64	45	106	170
Chancery Lane	191	119	66	21	26	170

Table 4.10 The most stable nodes with respect to their ranking position among the betweenness variants (standard and flow-based variants) and the most unstable nodes, for the London Transport dataset (transitive graph). $span(v)$ describes the nodes' span of rankings, i.e., the difference between its highest and its lowest ranking position among all centrality variants. In total, the network contains 268 nodes.

London Transport (transitive graph): Most stable nodes						
Node	standard	B_S	B_{SW}	B_{RW}	B_R	$span(v)$
King's Cross St. Pancras	1	1	1	1	1	0
Bank / Monument	2	2	2	5	5	3
Oxford Circus	5	3	5	2	3	3
Green Park	8	4	4	4	4	4
Baker Street	7	6	6	3	2	5
Wembley Park	34	34	32	28	28	6
Westminster	15	13	13	9	11	6
London Transport (transitive graph): Most unstable nodes						
Hammersmith (H&C)	168	251	254	66	36	218
Canary Wharf	241	38	49	52	43	203
Ickenham	49	204	199	243	250	201
Brixton	241	51	48	50	53	193
Vauxhall	241	50	64	65	51.5	191
Ruislip Manor	49	195	171	220.5	238	189
Pimlico	241	52	60	62	54	189
North Greenwich	241	55	70	71	57	186

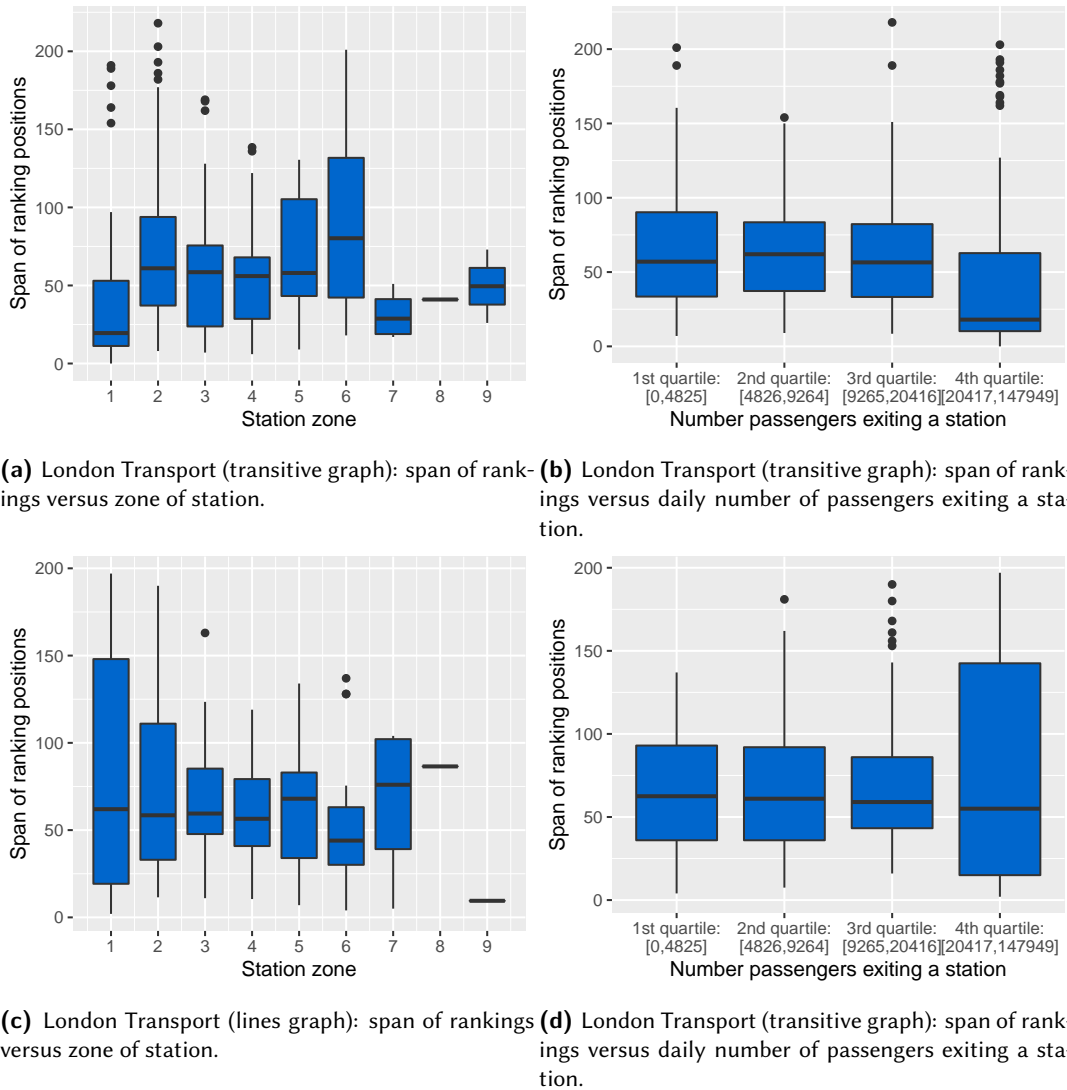


Figure 4.20 The nodes' span of ranking positions for the London Transport dataset (top: transitive graph; bottom: lines graph), plotted according to external node attributes. The plots on the left compare the nodes' span of ranking positions with the zone in which a station is located, the plots on the right compare the span of ranking positions with the average daily number of passengers exiting a station (both attributes provided by Transport for London [Tra17]).

Table 4.11 The most stable nodes with respect to their ranking position among the betweenness variants (standard and flow-based variants) and the most unstable nodes, for the Wikispeedia dataset. $span(v)$ describes the nodes' span of rankings, i.e., the difference between its highest and its lowest ranking position among all centrality variants. In total, the network contains 4589 nodes.

Wikispeedia: Most stable nodes						
Node	standard	B_S	B_{SW}	B_{RW}	B_R	$span(v)$
United States	1	1	1	1	1	0
United Kingdom	2	2	2	3	3	1
Europe	4	3	3	2	2	2
England	3	4	4	5	4	2
Africa	5	5	8	6	6	3
Earth	11	9	5	4	5	7
A Wrinkle in Time	4375.5	4383	4383	4379	4379	7.5
Abacá	4375.5	4383	4383	4379	4379	7.5
...	7.5
English language	10	13	18	17	14	8
World War II	7	7	16	7	8	9
Germany	6	6	12	9	15	9
France	14	10	25	13	12	15
Bird	23	32	36	22	38	16
China	16	14	19	28	32	18
India	15	17	11	24	31	20

Wikispeedia: Most unstable nodes						
Achilles tendon	4375.5	3908.5	632	917.5	2598	3743.5
History of Puerto Rico	187	3566	3636	3618.5	3725	3538
Matsuo Bashō	774	3955	4038	4071.5	4024.5	3297.5
Malwa (Madhya Pradesh)	1132	4162	4115	4379	4379	3247
Union Station (San Diego)	622	3747	3867	3864	3859.5	3245
Stock car (rail)	886	3976	4050	4071.5	4113.5	3227.5
Fauna of Australia	425	3559	3211	3330	3651	3226
Coupling (railway)	850	4028.5	3972	4000	4059	3209

large and diverse text corpus, the frequency of occurrence of each word is counted. Furthermore, they provide a score (called frequency rank) for each word, which is a function of its frequency and its dispersion [JCR64], where dispersion is a measure quantifying how evenly a word is distributed among the parts of the text corpus. For words contained in the network that are inflected forms (such as *are* is an inflected form of the verb *to be*), the score of the corresponding base form is used. For words that are contained several times in the frequency list (such as *use* as a noun and as verb), the entry with the higher score is kept. Not all words contained in the Wordmorph network are contained in the frequency list provided by [Dav8]. We used this frequency list and compared the words' frequency ranks to the span of ranking positions of the corresponding node. Figure 4.21a shows that there is no connection between the nodes' span of ranking position and the frequency rank of the corresponding word. The average span of ranking positions is approximately the same for all intervals of frequency ranks. A different plausible hypothesis is that common words are used more often by players when navigating from one word to another, and less common words are used less often. This would be reflected in the ranking by measure variants counting real trajectories, i.e., the measures B_R and especially B_{RW} . Figure 4.21b shows the ranking position of the nodes for all betweenness variants and their frequency rank from the text corpus. It can be observed that there is no relationship between these properties. For standard betweenness centrality, this

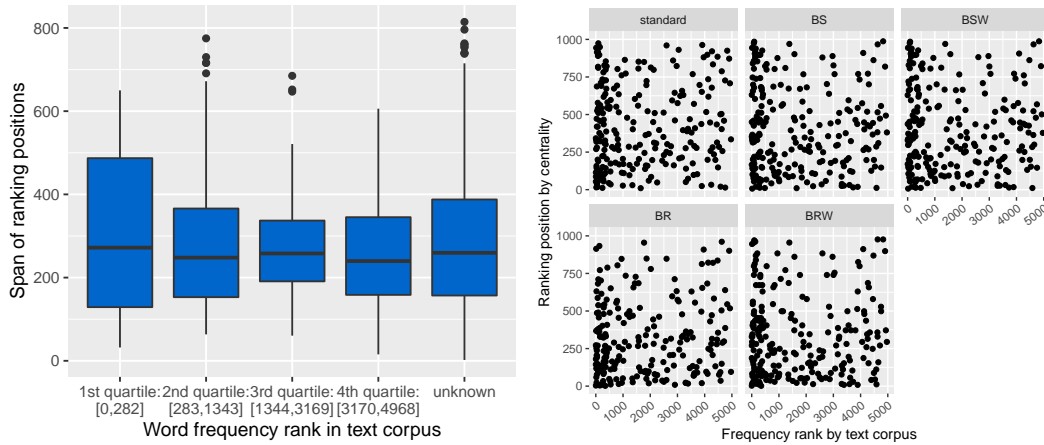
Table 4.12 The most stable nodes with respect to their ranking position among the betweenness variants (standard and flow-based variants) and the most unstable nodes, for the Wordmorph dataset. $span(v)$ describes the nodes' span of rankings, i.e., the difference between its highest and its lowest ranking position among all centrality variants. In total, the network contains 1008 nodes.

Wordmorph: Most stable nodes						
Node	standard	B_S	B_{SW}	B_{RW}	B_R	$span(v)$
ait	2	3	3	1	1	2
chi	992.5	991	991	996.5	992	5.5
ghi	992.5	1001	1001.5	1002	998	9.5
ais	3	2	2	4	12	10
ohm	992.5	1001	1001.5	1002	1003	10.5
urd	992.5	1005	1005	1005	1002	12.5
imp	992.5	996	996.5	991.5	984	12.5
ivy	992.5	1001	1001.5	1005	997	12.5
adz	992.5	996	996.5	999.5	986	13.5
oxy	992.5	1001	996.5	1002	1006	13.5
Wordmorph: Most unstable nodes						
Node	standard	B_S	B_{SW}	B_{RW}	B_R	$span(v)$
aah	141	834	838	955.5	932	814.5
hmm	130	740	724	814	926	796
arm	538	822	823	82.5	48	775
eta	719	966	966	839	203	763
phi	123	859	859	883.5	882	760.5
ahi	124	790	793	883.5	816	759.5
eld	767	980	980	861	226	754
oho	45	707	697	798.5	415	753.5
ute	539	923	924	699.5	184	740
ope	144	783	780	883.5	626	739.5

is not surprising: There is no reason why the structure of a network where an edge corresponds to a Hamming distance of 1 should reflect the frequency of occurrence of words. For the measure B_{RW} , this would have been plausible, but it turned out that there is no connection between the occurrence of a word in the text corpus and the usage of the corresponding node by players playing the Wordmorph game.

4.7 A note on the results' sensitivity to changes of the trajectory set

The results of all introduced flow-based centrality measures are heavily dependent on the available trajectory set \mathcal{P} . Since the trajectory set \mathcal{P} is based on empirical observations, it is possible that there are errors or uncertainties in the data. Furthermore, in order to obtain \mathcal{P} , usually preprocessing steps are necessary where different researchers might make different decisions leading to different trajectory sets \mathcal{P} . For the Wikipedia dataset used in this work, we decided to exclude unsolved solution attempts as well as players' solutions that are longer than 30 steps. The reason for this decision is the assumption that such long paths are not the result of thoughtful playing, but rather of untargeted clicking, possibly for a different purpose than solving the task. This is certainly a decision that can be made differently. The length restriction excludes only 45 solutions (out of more than 51000). The effect of this exclusion on the results of the analysis was however surprisingly large: The betweenness variant B_R was extremely sensitive to such preprocessing deci-



(a) The node's span of ranking positions for the be- (b) The node's ranking position for each flow-based tweenness variants shown against its frequency rank. betweenness centrality shown against the word's frequency rank.

Figure 4.21 For the Wordmorph dataset, we used the word frequencies in a large text corpus provided by [Dav8] as a comparison variable. The frequency rank for a word is a score dependent on its frequency of occurrence and its dispersion in the text corpus (see text for explanation). Note that not all nodes of the dataset are contained in the word frequency list of [Dav8].

sions. In addition to the analysis of the previous sections, we computed all flow-based betweenness variants on the basis of the unrestricted trajectory set of Wikispeedia, i.e., \mathcal{P} is the set of players successful solution attempts, without any length restriction.

Table 4.13 shows the Spearman rank correlation coefficient of the flow-based betweenness variants to standard betweenness centrality, based on the unfiltered trajectory set and (as before) based on the trajectory set where the 45 long solutions were excluded. It can be seen that no effect is visible on the variants B_S , B_{SW} , and B_{RW} , but for the variant B_R , the correlation coefficient drops from 0.3 to 0.39 by only excluding 45 trajectories.

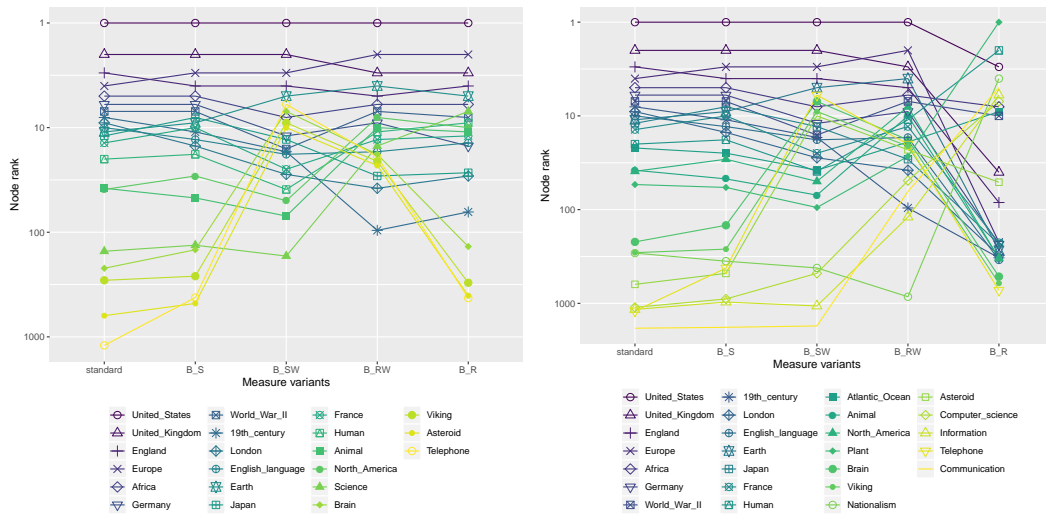
Figure 4.22 shows the effect on the top most central nodes: As in Figure 4.11, for all nodes that are among the ten most central nodes with respect to at least one betweenness variant, their ranking positions for all variants are shown. To compare the results, the results for both dataset versions are shown, once for the dataset as used in the previous sections, once for the dataset where long paths were not filtered out. Also here, the sensitivity of the variant B_R becomes obvious: While the nodes' ranking behavior for all other variants is rather similar for both dataset versions and the top ten nodes show a large overlap, there is a clear discrepancy for the variant B_R . This is due to the definition of B_R : The flow-based count of paths, i.e., the definition of $\sigma_{st}^{\mathcal{P}}(v)$ only requires that the nodes s , v , and t occur in the same real trajectory (in this order) to be counted. This explains the large impact of long trajectories: Each triple of nodes (s, v, t) contained in the trajectory in this order contributes to the corresponding $\sigma^{\mathcal{P}}$ value. Since these long trajectories were not targeted at all (the longest trajectory contains 404 steps although the target could have been reached within 2 steps!), the values of $\sigma_{st}^{\mathcal{P}}(v)$ change considerably when these long trajectories are included in \mathcal{P} . This explains the considerably different ranking behavior of B_R when including or excluding the long trajectories.

On the other hand, the remaining flow-based betweenness variants and all closeness variants (results not shown here) do not show this effect. It has not been shown yet how robust they are against major changes in the underlying trajectory set \mathcal{P} , but they show at least less sensitivity to changes in \mathcal{P} than B_R .

Table 4.13 Effect of preprocessing decisions on the results of flow-based betweenness centralities: Spearman rank correlation coefficient and weighted overlap of all flow-based variants compared to standard centrality, once for the dataset Wikispeedia as used in the previous sections, once for a slightly modified dataset, where 45 trajectories with more than 30 steps that were not removed from \mathcal{P} before the flow-based betweenness variants were computed with them.

Dataset version	Spearman correlation			
	B_S	B_{SW}	B_R	B_{RW}
Wikispeedia: all trajectories	0.83*	0.81*	0.78*	0.79*
Wikispeedia: trajectories with > 30 steps filtered out	0.83*	0.81*	0.81*	0.79*

Dataset version	Weighted overlap τ_w			
	B_S	B_{SW}	B_R	B_{RW}
Wikispeedia: all trajectories	0.21	0.25	0.39	0.31
Wikispeedia: trajectories with > 30 steps filtered out	0.21	0.25	0.30	0.31



(a) Wikispeedia: trajectories with more than 30 steps were excluded from \mathcal{P} . (b) Wikispeedia: \mathcal{P} contains all solving solutions.

Figure 4.22 Effect of preprocessing steps on the results of flow-based betweenness measures: The figure shows the ranking behavior of those nodes that are among the ten highest ranked nodes with respect to at least one betweenness variant (as in Figure 4.11). On the left: for the dataset as used in the previous sections (the figure is the same as in Figure 4.11b) where 45 solutions of a length of more than 30 steps were removed from \mathcal{P} before the flow-based betweenness variants were computed. On the right: all solving solutions are included in \mathcal{P} .

4.8 Which assumptions matter, which don't?

In the previous sections, the flow-based centrality measures were compared to the corresponding standard centrality measure, i.e., standard betweenness and standard closeness centrality. In order to investigate whether the violation of certain assumptions contained in these standard measures has a larger impact on the results than the violation of other assumptions, we will compare the flow-based centrality variants among each other in this section. For an appropriate interpretation of the results, we not only need to know *that* an assumption is violated by the real-world network flow contained in a dataset, but we need a measure that quantifies *to which extent* a certain assumption is not met by the real network flow. Otherwise, the results of the different datasets cannot be compared. We will therefore introduce measures for each single assumption which measure how far the real network flow is from the assumed process model with respect to a specific assumption¹⁰:

There is flow between each node pair We have shown that for none of the datasets, the statement that flow is observed between each node pair holds. In order to quantify this observation, we simply used the percentage of node pairs between which at least one real trajectory was observed in the dataset, relative to the total number of all node pairs in the graph. Table 4.14 shows the corresponding values for the datasets we used: While we assumed 100 % for the assumed process model, for the transportation datasets, approximately half of all node pairs were used as source and target of any observed trajectory. For the game datasets, Wikispeedia and Wordmorph, this percentage was much smaller, 1.1 % and 0.13 %. However, it needs to be noted that the corresponding networks are larger than the networks of London Transport and DB1B, so that a higher number of trajectories is needed to ensure at least one trajectory between each node pair. For Wikispeedia, for example, given the available number of trajectories and the network, a percentage value of 0.24 % is the maximal value that could be reached. For Wordmorph, only 1.1 % of all node pairs were the source and target of at least one observed trajectory; however, with 11651 trajectories contained in the dataset, a percentage of only 1.14 % is actually possible.

Equal amount of flow between each node pair Furthermore, we assumed that the amount of flow between each node pair is equal. In order to quantify to which extent this was not the case, we considered those node pairs between which there exists at least one trajectory. An frequently used measure for quantifying the inequality of a distribution is the Gini coefficient, which was introduced in Chapter 3. Table 4.14 shows the resulting values. It can be seen that the Wordmorph dataset shows the smallest value, even close to 0, while DB1B shows the largest value, even close to 1 indicating that the amount of flow between the node pairs used is distributed in a very imbalanced way for DB1B.

Usage of shortest paths In order to measure to which extent the real-world network flow is using shortest paths, we used two measures: We first considered the percentage of observed trajectories whose length is equal to the length of the shortest path from its source to its destination. These values are quite high (above 69 %) for all datasets except Wikispeedia where it is only 22 %. We furthermore computed the factor by which a trajectory was longer than its corresponding shortest path, and computed the average of those values (including those that were as short as possible). Table 4.14 shows that the trajectories of the London Transport dataset are generally very close to the optimum, while the trajectories of Wikispeedia are, on average, longer than the shortest path by a factor of 1.76. With 1.27, the value for DB1B seems surprisingly high. This is due to the fact that the majority of DB1B trajectories have a length of 1 or 2, and there are a large number of trajectories with length 2 and a shortest path of length 1. All these trajectories contribute to the overall value with a value of 2.

With a quantification at hand regarding the extent to which the real-world network flows show

¹⁰Since all considered datasets contain the flow of transfer processes, this assumption does not need to be considered here.

Table 4.14 Violation of assumptions in the considered datasets: For each assumption, a corresponding measure was used to quantify *to which extent* the real-world network flow deviates from its assumed behavior.

Assumption: There is flow between each node pair				
<i>Measured by: Percentage of node pairs used</i>				
assumed	DB1B	London Transport	Wikispeedia	Wordmorph
100 %	51 %	46 %	0.13 %	1.1 %
Assumption: Equal amount of flow between node pairs used				
<i>Measured by: Gini coefficient $\in [0, 1]$</i>				
assumed	DB1B	London Transport	Wikispeedia	Wordmorph
0	0.95	0.73	0.38	0.0086
Assumption: Usage of shortest paths				
<i>Measured by: Percentage of trajectories with optimal length</i>				
assumed	DB1B	London Transport	Wikispeedia	Wordmorph
100 %	69 %	89 %	22 %	71 %
<i>Measured by: Factor by which trajectories were longer than optimum</i>				
assumed	DB1B	London Transport	Wikispeedia	Wordmorph
1.0	1.27	1.02	1.76	1.22

Table 4.15 The tables contain the Spearman rank correlation coefficient between the values of the flow-based betweenness measures (white cells, upper right cells), and the weighted overlap τ_w between the rankings of the measures (gray cells, lower left cells). Starred values for the Spearman rank correlation coefficient indicate a p -value of ≤ 0.05 .

	B_S	B_{SW}	B_{RW}	B_R
B_S		0.94*	0.94*	0.91*
B_{SW}	0.16		1.00*	0.97*
B_{RW}	0.15	0.02		0.97*
B_R	0.15	0.15	0.14	

(a) DB1B

	B_S	B_{SW}	B_{RW}	B_R
B_S		0.98*	0.95*	0.91*
B_{SW}	0.11		0.96*	0.91*
B_{RW}	0.18	0.16		0.97*
B_R	0.23	0.24	0.12	

(b) Wikispeedia

	B_S	B_{SW}	B_{RW}	B_R
B_S		0.92*	0.75*	0.78*
B_{SW}	0.19		0.89*	0.85*
B_{RW}	0.35	0.21		0.95*
B_R	0.29	0.26	0.18	

(c) London Transport (lines graph)

	B_S	B_{SW}	B_{RW}	B_R
B_S		0.97*	0.94*	0.96*
B_{SW}	0.09		0.96*	0.92*
B_{RW}	0.17	0.13		0.97*
B_R	0.15	0.20	0.10	

(d) London Transport (transitive graph)

	B_S	B_{SW}	B_{RW}	B_R
B_S		1.00*	0.91*	0.60*
B_{SW}	0.02		0.91*	0.60*
B_{RW}	0.25	0.25		0.77*
B_R	0.48	0.48	0.33	

(e) Wordmorph

different properties than the assumed process model, it is interesting to consider the flow-based centrality variants and compare their results among each other. For each dataset and each two flow-based centrality measures (separately for betweenness and closeness variants), the Spearman rank correlation coefficient and the weighted overlap between their values and rankings were computed (see Section 4.3 for details). Table 4.15 shows the results for the betweenness variants. Several observations can be made:

- For all datasets, the correlation between the weighted and unweighted variants, i.e., between B_S and B_{SW} as well as between B_R and B_{RW} , is high, indicating that incorporating the amount of flow between nodes as a weight into the measure, does not have a large impact on the resulting ranking. This is not surprising for the Wordmorph dataset where the amount of flow between all node pairs used is 1 for almost all node pairs and the Gini coefficient is close to 0. For Wikispeedia, the higher amount of flow between certain node pairs is mainly due to the increased frequency of four node pairs (intended by the researchers, see Section 4.6.1 and [WL12b]), which explains that the overall effect on the remaining nodes was minor. For DB1B, where the assumption of equal amount of flow is also not met, the correlation is still high with 0.94. However, the weighted overlap for B_S and B_{SW} is the largest among all variant combinations within the DB1B dataset.
- When considering the correlations between the variants counting shortest paths and the variants counting real trajectories, the results are different for the different datasets. For the Wordmorph dataset, the correlation coefficients are the lowest, dropping to 0.60 for the variants B_S and B_R , and for the variants B_{SW} and B_{RW} (note that for Wordmorph, the weighted and unweighted variants are almost identical because the assumption of equal amount of flow between the nodes used is approximately satisfied here). The weighted overlap even increases to the value of 0.48. Nevertheless, even for this dataset, the correlation is still positive and far from 0. Considering the transportation datasets DB1B and London Transport, it can be observed that although according to Table 4.14, the assumption of shortest paths is less satisfied for DB1B than for London Transport, the correlations are opposite: The correlations between B_{SW} and B_{RW} , and between B_S and B_R , are smaller for London Transport than for DB1B.

4.9 Summary and limitations

4.9.1 Limitations

There are limitations of this work that need to be mentioned. As for any data-based analysis, the results are highly dependent on the data collection methods and all data preprocessing steps. In this work, we used datasets consisting of a network representation of the system and a set of trajectories. For one dataset, both the network representation and the set of trajectories were built from the same data; for the remaining datasets, they were built from different sources. In either case, it is neither guaranteed that the data we used is complete nor that it is free of any errors. As in many cases, the network representation of the system needs to be defined by the researcher, and there are several plausible choices for this. Other network representation choices would probably have yielded different results. For the London Transport dataset, there are two graph representations that are both plausible for the application of the considered centrality indices, so we performed the same analysis on both representations. For the preprocessing of the trajectory set of each dataset, different decisions are also possible and plausible. For the game datasets for example, a plausible filtering step is to exclude all trajectories that are longer than a certain threshold. In Section 4.7, we demonstrate that one of the introduced flow-based betweenness centralities is sensitive to this decision. In future work, a systematic analysis of the sensitivity of the introduced measures to the input trajectory set \mathcal{P} is needed.

To test the impact of single assumptions on the results of centrality indices, we decided to use

information from real-world network flows. A different approach would be to use synthetic data (or solving it analytically), which would give a higher controllability of the data and make it easier to consider the assumptions separately. However, we chose to use the data of real-world network flows in order to demonstrate that there is also an effect for networks with processes for which the application of the classic centrality indices is reasonable according to Borgatti's classification [Bor05]: We intentionally chose datasets containing transfer processes where each process entity has a target that it tries to reach on the shortest path. If synthetic data containing trajectories satisfying all but one assumption were used, the effect on the process-driven centrality measures would not be surprising. However, we even found an effect of the violation of assumptions on the centrality results when using datasets that—in theory—could satisfy the assumptions.

A further limitation of the following analysis is the bandwidth of the datasets used. We applied our analysis to only four datasets of which two are transportation systems and two are games. To improve the validity of the results, a larger set of datasets from more different domains would be necessary. Unfortunately, only very few datasets satisfying the requirements for this analysis are available which is why the usage of more diverse datasets was not possible.

4.9.2 Summary

Many centrality indices implicitly contain a process model with certain properties. The ranking induced by a centrality index can then be interpreted as the nodes' importance with respect to a process with these properties. For example, betweenness centrality measures a node's importance with respect to a network process consisting of indivisible entities traveling on shortest paths through the network. In the previous chapter (Chapter 3), we demonstrated that the behavior of real-world network flows is not necessarily in conformity with the assumed properties. In this chapter, we considered whether this has an impact on the results of the centrality indices. For this reason, different *flow-based variants* of betweenness and closeness centrality were introduced which, for each assumption, either used the "ideal" process model or the properties of the real-world network flow. This technique allows us to separate between the different assumptions contained in the centrality indices.

When applying these variants on four different datasets, two containing passenger flow in transportation systems and two containing player flows in game networks, we found that violation of the assumptions by the real-world network flow does have an effect on the results of the centrality index. When comparing the rankings of the flow-based variants to the corresponding standard centrality index, we found that their correlation was still high in general. However, when considering which nodes are impacted by replacing the process model with the behavior of the real-world flow, we observed nodes whose ranking positions changed enormously. We even found considerable ranking deviations among the highest-ranked nodes which is remarkable since, when centrality indices are applied, usually only the highest-ranked nodes are of interest. If even the ten most central nodes are different for the flow-based variants and their corresponding standard centrality, the interpretability of any centrality-based result becomes more difficult.

We further analyzed which nodes are impacted more by incorporating the properties of the real-world network flow. For the DB1B dataset containing passenger flow in the US airline transportation system, we found that particularly the ranking position of small airports was affected by incorporating the properties of the real-world network flow while for large airports, such as New York or Los Angeles, the assumed flow and the actual network flow appeared to be in accordance such that their ranking position was stable among all variants and standard centrality.

Furthermore, in order to investigate whether the violation of certain assumptions has a larger impact on the results than the violation of others, we compared the flow-based centrality variants among each other. In general, we found that their correlations were rather high for all datasets

although the corresponding assumptions were not necessarily met by the real-world flow contained in the respective dataset. Nevertheless, we also found cases where the incorporation of real trajectories into the betweenness instead of shortest paths leads to considerable variations in the rankings.

Although the assumptions of centrality indices are well-known in the community, to the best of our knowledge, this is the first work that systematically analyzes the impact of single assumptions by using real-world datasets.

Summarizing trajectories of real-world network flows

Chapter outline

In this chapter, several distance and similarity measures for walks in a graph are derived. We are not aware of any work proposing a distance or similarity measure for walks. Therefore, we review data structures as which a walk can be modeled: as a set, as a sequence, as a set of elements in which each two elements have a distance to each other (thus as a set of points in a metric space), and as a sequence of elements with a pairwise distance between them (thus as a polygonal curve in a metric space). For each of those structures, similarity or distance measures have been proposed. In this chapter, we review the existing measures and adapt them to the application on walks in a graph. These similarity and distance measures are then used to cluster the empirically observed walks of a dataset into groups. We use a dataset containing the state space of a board game as network, and humans trying to solve the game as walks through the state space. The main goal is to evaluate the proposed similarity measures whether they are able to distinguish between structurally different walks.

This chapter is based on the work published in [3] (for similarity and distance measures) and (for clustering trajectories) based on the publication [2]^{ab}.

^aIn the publication [2], the additive version of the (simplified) additive discrete Fréchet distance is called CoMapPa2 (CoMapPa1) because we were not aware that this measure has been proposed before.

^bIn a previous work [Boc15], a preliminary set of similarity and distance measures was proposed, which is extended in the present work.

5.1 Motivation

The previous chapters revealed that the properties of real-world network flow cannot be reproduced by simple trajectory models, i.e., shortest paths and random walks. A similar situation was described by Watts and Strogatz in their famous small-world article [WS98]: Networks deduced from real-world systems showed neither the structural properties of regular graphs nor those of random graphs. They therefore introduced a new model for generating graphs (*small world model*) whose properties could be scaled with a parameter between the two extremes regular graphs and random graphs.

When considering real-world network flows, there is no such model available that captures the properties of an empirical network flow. Until such a realistic model is available, it is necessary to consider the empirically collected trajectories instead. However, given a large set of trajectories, it is often not feasible to consider each single trajectory, so a grouping of the trajectories into subsets of trajectories with similar structural properties is helpful. First, considering representative

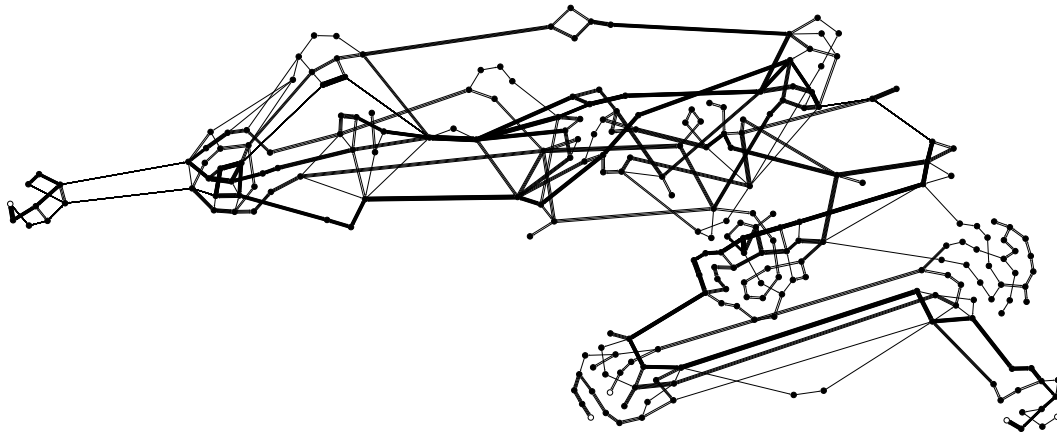


Figure 5.1 A state space of a board game and human navigation in it when solving the game. All players start in the node leftmost drawn and aim at reaching the solution state drawn in the bottom right corner.

trajectories for each group instead of the complete trajectory set reduces computational complexity. Second, identifying groups of structurally similar trajectories might help to develop a more realistic model for trajectories.

Consider Figure 5.1 which shows the state space of the single-player board game Rush Hour (as introduced in Chapter 3) where a node represents a game configuration and edges represent valid game moves. Attempts of humans trying to solve the puzzle can be understood as walks through the state space. The edges in the figure are drawn bolder if more players used them while solving the game. For the human eye, it is easy to recognize that most players take “somehow similar” ways to solve the game. With respect to this application scenario, it would be helpful to identify these similar solutions, for example in order to identify different solution strategies used by the players.

Finding meaningful groups of any type of trajectories is also relevant in other areas of research:

- Consider an e-learning platform or an online course providing learning material to students (videos, documents, tests, etc). A student taking a course and using the provided material items creates a sequence of items, depending on the order and frequency of use of the material items. Grouping these trajectories into meaningful groups could be used to identify different learner types and help to improve learning efficiency by providing different materials (or the same material items in a different order) to students of different learner types.
- Similarly, consider the purchase history of users in a commercial context. For each user, there is a sequence containing which product the user has bought in which order. Finding groups of similar purchase histories might lead to better product recommendations [YML⁺14].
- Trajectories representing solutions of humans trying to solve a puzzle can be clustered in order to identify different strategies for solving the puzzle.
- Sequences of proteins can be grouped according to the assumption that protein sequences within the same group have a similar biological function [YW03].
- Trajectories from tracking moving objects can be clustered for various purposes: predicting future object behavior [SB00], identifying trajectories of the same object [VKG02, AT06], identifying leaders and followers [AGLW07], classifying the shopping behavior of customers in a supermarket by considering their routes through the store [LBF05, STYS16], or identifying similar strategies of football teams by analyzing the players’ movements in a game [GH17].

In order to achieve a meaningful grouping, a suitable similarity measure for walks is needed. To the best of our knowledge, no similarity measure dedicated particularly to walks in graphs exists. There

are, however, dozens of different similarity and distance measures for other data structures from various areas of research. In applications such as video surveillance systems, the positions of moving objects are tracked through consecutive frames and their trajectory is extracted. For automatically distinguishing between regular and anomalous trajectories, similarity measures between these trajectories have been developed [VKG02, ME02, BSK04, BKS03]. The most frequently used similarity measures for this scenario are the length of the longest common subsequence [BSK04, VKG02] and the Hausdorff distance [JJS04]. Other trajectories of moving objects are tracked by GPS sensors, for example animals in the wild [SBvLP⁺11], routes of cyclists [VHL⁺14], or trajectories of pedestrians [AT06]. In this area of research, the discrete Fréchet distance [Fré06, EM94] is often used, for example to detect recurring subtrajectories [BBG⁺11]. Other approaches consider sequences of events and aim at identifying similar sequences where variants of edit distances or alignment methods from string matching [Gus97] are often used, for example for measuring the similarity between event sequences [MR97].

All these approaches from different areas of research exist and they can all be adapted to walks in graphs. The main questions posed in this chapter are therefore:

How can the similarity of two walks in a graph be measured? Which information contained in a walk is essential such that it needs to be incorporated into a similarity measure for walks? Which similarity or distance measure is best suited to find meaningful groups of trajectories?

This chapter consists of two parts: We will first introduce and review existing similarity measures for the following four data structures: sets, sequences, sets of points in metric space, and polygonal curves in metric space. Second, we will test which of the introduced measures is best suited for finding meaningful groups of trajectories, by using a dataset with ground truth. For the sake of simplicity, in this chapter, the underlying graph is assumed to be undirected, unweighted, and simple.

5.2 Similarity and distance measures for walks in graphs

In general, a similarity measure is a function $s : X \times X \rightarrow \mathbb{R}$ which indicates for two elements of a set X how similar they are. Usually, a high value of s implies high similarity of the two objects (according the criteria used by s). A distance measure on a set X is a function $\delta : X \times X \rightarrow \mathbb{R}$ that indicates the distance or dissimilarity of two elements of X . A low value of δ implies high similarity. There are several desirable properties of distance measures:

Definition 5.1: Properties of distance and similarity measures

Non-negativity A distance measure is said to satisfy *non-negativity* if for all $x, y \in X$, it holds that $\delta(x, y) \geq 0$.

Coincidence A distance measure δ is said to satisfy *coincidence* if for all $x, y \in X$, it holds that $\delta(x, y) = 0 \Leftrightarrow x = y$. Hence, from a distance value of 0, it can be deduced that the two elements are identical, and vice versa.

Symmetry A distance measure satisfies *symmetry* if for all $x, y \in X$, it holds that $\delta(x, y) = \delta(y, x)$.

Triangle inequality A further desirable property is the *triangle inequality*: for all $x, y, z \in X$, the following inequality holds: $\delta(x, z) \leq \delta(x, y) + \delta(y, z)$.

Boundedness A distance measure δ or similarity measure s is *bounded above* by a constant $C \in \mathbb{N}$ if for any two elements $x, y \in X$, it holds that $\delta(x, y) \leq C$ and $s(x, y) \leq C$,

respectively.

A distance measure satisfying non-negativity, coincidence, symmetry, and the triangle inequality is called a *distance metric*.

If a similarity measure is bounded, we can formulate the above properties also for similarity measures:

Non-negativity A similarity measure is said to satisfy *non-negativity* if $s(x, y) \geq 0$.

Symmetry A similarity measure is said to satisfy *symmetry* if for all $x, y \in X$, it holds that $s(x, y) = s(y, x)$.

Coincidence A similarity measure is said to satisfy *coincidence* if for all $x, y \in X$, it holds that $s(x, y) = C \Leftrightarrow x = y$.

Triangle inequality A similarity measure is said to satisfy the *triangle inequality* if for all $x, y, z \in X$, it holds that $s(x, z) \geq s(x, y) + s(y, z) - C$.

We claim that there are three kinds of information contained in a walk (see also Figure 5.2):

- (i) Information on which elements are contained in a walk, i.e., which nodes and edges.
- (ii) Information on the order in which they occur in the walk.
- (iii) Information on where in the graph the walk is located.

From these three kinds of information, corresponding similarity measures can be derived: Similarity measures using the first kind of information consider a walk as a set of elements, so, similarity measures for sets can be used. Similarity measures using the order of the elements consider the walk as a sequence of nodes and edges, so, existing similarity measures for sequences or strings can be used. If the position of the walk nodes within the graph is of interest (and only of those), we consider the position of the single walk nodes within the graph. Hence, a walk is considered as a set of points in a metric space and existing similarity measures for this kind of data structure can be used. If all three kinds of information—elements, order, and position in the graph—are to be considered, a walk can be modeled as a discrete curve in a metric space. This data structure allows considering all three kinds of information and there exist similarity measures for this type of structure.

5.2.1 A walk as a set

The simplest approach for modeling a walk is to only consider the nodes or edges it contains as a set. For the comparison of two sets, basic measures are available, for example the number of common elements, or as a normalized variant, the Jaccard index [Jac12]: For two sets A and B , the Jaccard index is defined as

$$s_{Jaccard} = \frac{|A \cap B|}{|A \cup B|}.$$

Node and edge set similarity

For a walk, we can either consider the nodes or the edges contained in it as the relevant set¹.

Definition 5.2: Node and edge set similarity for walks

For walks P and Q , their node set and edge set similarity are defined as follows. **Node set similarity:**

$$s_{nss}(P, Q) = |V(P) \cap V(Q)| \quad (5.1)$$

¹These two similarity measures are also contained in previous work [Boc15]

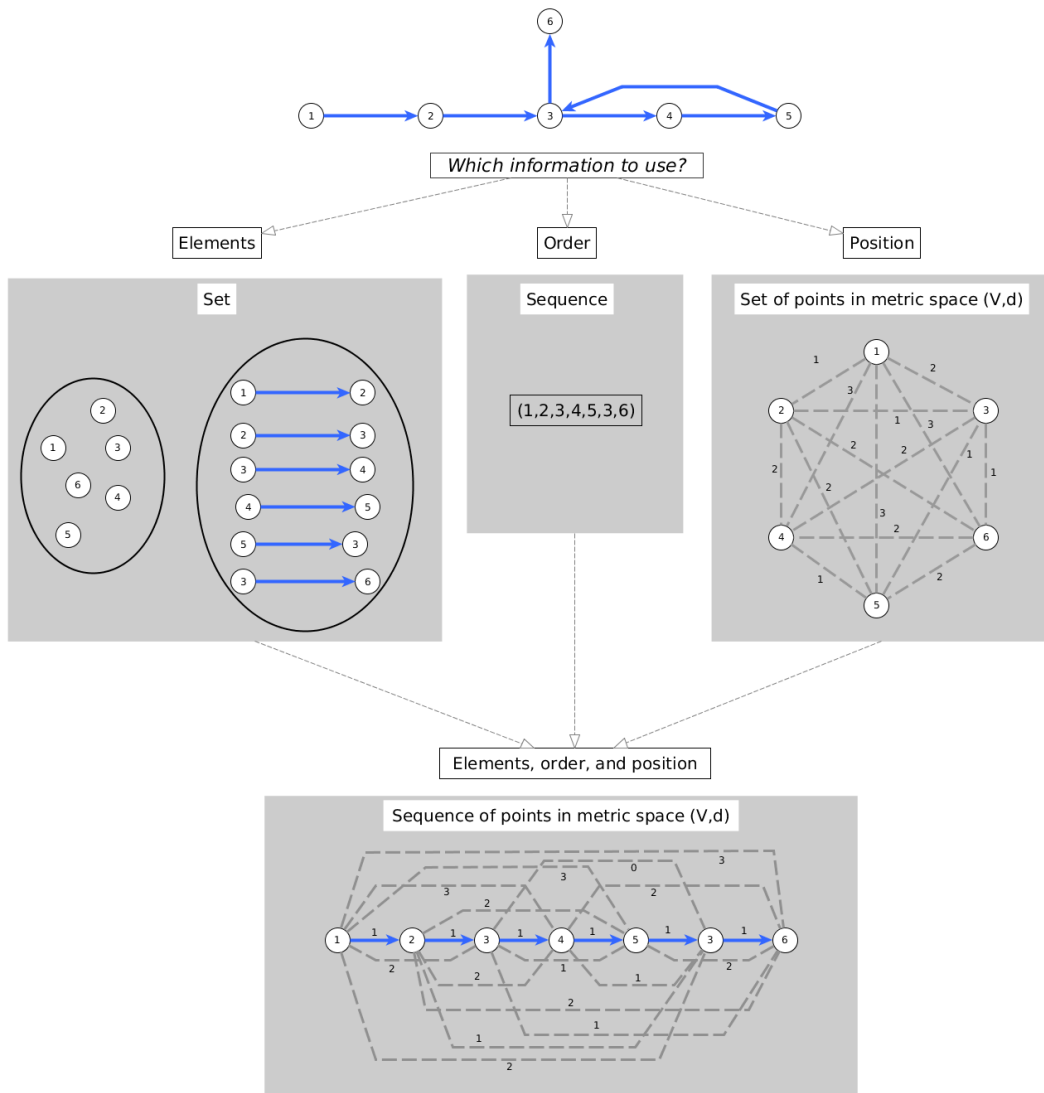


Figure 5.2 An overview of how a walk in a graph can be modeled, depending on which kind of information is used. When only its contained elements are of interest, a walk can be modeled as a set (of nodes or edges); when its order is of interest, it can be modeled as a sequence; when its position in the graph is of interest, it can be modeled as a set of points in the metric space (V, d) where V is the set of nodes in the graph and d is the graph distance between the nodes (indicated by the gray dashed edges with labels). An approach using all three kinds of information is modeling as a sequence of points in the metric space (V, d) , i.e., as a polygonal chain or polygonal curve.

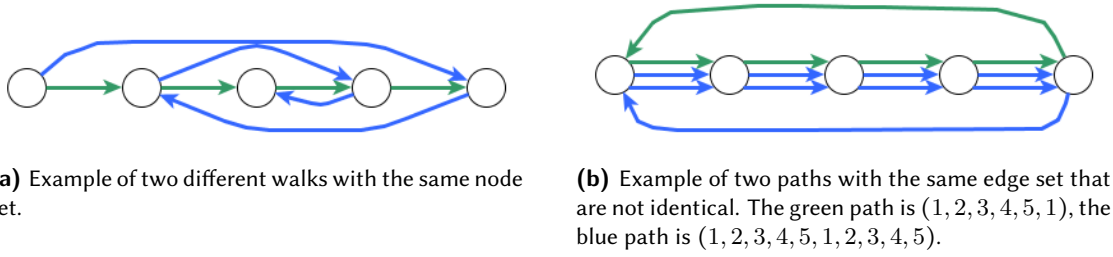


Figure 5.3 Examples of walks where the set-based measures are not satisfactory.

Normalized node set similarity:

$$s_{nss}^N(P, Q) = \frac{|V(P) \cap V(Q)|}{|V(P) \cup V(Q)|} \quad (5.2)$$

Edge set similarity

$$s_{ess}(P, Q) = |E(P) \cap E(Q)| \quad (5.3)$$

Normalized edge set similarity

$$s_{ess}^N(P, Q) = \frac{|E(P) \cap E(Q)|}{|E(P) \cup E(Q)|} \quad (5.4)$$

Properties of the set similarities These measures are easy to compute and satisfy boundedness (bounded by $|V|$, $|E|$, and 1, respectively), non-negativity, symmetry, and the triangle inequality [Kos19]. However, it is obvious that a lot of information contained in a walk is not considered, if a walk is broken down into a set of nodes or edges. This is why the introduced measures do not satisfy coincidence: There are walks, such as those shown in Figure 5.3, which yield the highest possible similarity measure, but are not identical.

5.2.2 A walk as a sequence

When a walk is considered as a sequence of nodes or a sequence of edges, the order of the contained elements is taken into account and similarity measures designed for sequences or strings can be applied. In the area of bioinformatics, several measures have been designed to compare genome sequences. Interestingly, most existing measures can be formulated as edit distances [Lev66]: Assume a set of allowed edit operations, such as inserting, deleting, or substituting an element in the string, where each operation is associated with a cost. The edit distance between two sequences is then the minimal cost necessary for transforming the one sequence into the other by using the allowed edit operations.

Longest common subsequence measure

When inserting and deleting are allowed edit operations, and both operations are assigned a cost of 1, we obtain a measure that is also called the longest common subsequence distance [NW70]: For a sequence $a = (a_1, \dots, a_l)$, a subsequence of a is defined as any sequence of elements that can be obtained from a by deleting elements. Note that the elements of a subsequence of a do not necessarily occur consecutively in a . Thus, (a, c, g) is a subsequence of the sequence (a, b, c, d, e, f, g, h) . For two sequences a and b , let $lcs(a, b)$ denote the length of the longest subsequence of a and b . Let $lcs(P, Q)$ for walks P and Q be defined accordingly based on the node sequences of P and Q .

Definition 5.3: LCS similarity for walks

For two walks P and Q as sequences of nodes, we define their **LCS similarity**^a as:

$$s_{lcs}(P, Q) = lcs(P, Q) \quad (5.5)$$

and their **normalized LCS similarity** as

$$s_{lcs}^N(P, Q) = \frac{lcs(P, Q)}{\max\{|P|, |Q|\} + 1}. \quad (5.6)$$

^aThis similarity measure is contained in a previous work [Boc15].

Properties of the LCS similarity These measure variants have several desirable properties, such as non-negativity, symmetry, coincidence (at least the normalized version), and the triangle inequality [Nav01, ZYPW02]. Note that the unnormalized version does not satisfy boundedness. However, there are several drawbacks for the case of walks: First, the concept of edit distances is not natural for walks. When applying an edit operation that is reasonable for strings or sequences, on a sequence representing a walk, this might yield a sequence that is no longer a possible walk in the graph. Second, this type of distance measures is designed for comparing sequences whose length is large compared to the size of the alphabet from which the letters are taken. This is the case, for example, for genetic sequences consisting of only four types of letters while containing thousands of letters. This is different for walks where a walk usually only contains a small subset of the nodes in the graph. Hence, when comparing two walks in a graph, they usually only contain few common nodes for which the order can be taken into account by the LCS similarity. Even worse, for walks with no common nodes, the similarities introduced so far will yield a value of 0 no matter what they look like. Our intuition, however, is that two walks that somehow "run parallel" through the graph should be regarded as more similar—even though they do not share any node—than two walks that are structurally totally different from each other—even though they share a few nodes. In order to quantify this, the position of the contained nodes in the graph needs to be considered.

5.2.3 A walk as a set of points in metric space

In order to take into account the position of the nodes of a walk within the graph, a natural choice is to consider the graph distances of the nodes of the walk. We will first introduce several distance measures for walks that *only* incorporate the graph distance of the walk nodes between each other, but not their order in the walks. Thus, a walk is modeled as a set of nodes where each two nodes have a distance to each other. This can be formulated as a set of points in a metric space: a metric space is defined as a (X, δ) with a set X and a distance metric on X . In this case, the set of graph nodes V together with the graph distance d forms a metric space (V, d) (if G is connected and undirected). A walk can then be understood as a set of points in (V, d) by only considering its nodes in the walk and their embedding in the graph. For measuring the distance between two sets of points in a metric space, there exist several distance measures.

The Hausdorff distance

Hausdorff introduced a metric that takes into account the maximal distance between points in the sets, which is named as Hausdorff distance. More precisely, for each point in the one set, the distance to its closest point in the other set is computed: the Hausdorff metric is then the maximum of these distances. We can easily adapt this measure for the application to walks in a graph and get the following definition:

Table 5.1 Similarity and distance measures for paths, depending on how a path is modeled. The references either refer to the authors who introduced the original measure or to authors who applied it in the corresponding context.

Modeling as	Measure	References	Used for walks as
Set	Jaccard Index	[Jac12]	$\frac{s_{nss}^{(N)}}{s_{ess}^{(N)}}$
Sequence	longest common substring distance	[Gus97]	
	longest common subsequence distance	[NW70]	$s_{lcs}^{(N)}$
	Levenshtein distance	[Lev66]	
	further edit distances	[Nav01]	
Set of points in metric space	Hausdorff distance	[Hau14]	δ_h
	matched average distance	[Boc15]	$\delta_{mad}^{(N)}$
	sum of minimum distances	[Nii87]	
	(Fair) Surjection distance	[Odd86]	
	Link distance	[EM97]	
	Matching distance	[RB01]	
Sequence of points in metric space	Fréchet distance	[Fré06, AG95]	
	discrete Fréchet distance	[EM94]	δ_{dF}
	additive discrete Fréchet distance	[EM94]	$\frac{\delta_{adF}^{(N)}}{\delta_{sadF}^{(N)}}$
	LCSS distance	[VKG02]	

Definition 5.4: Hausdorff distance for walks

For two walks P and Q in graph G , their **Hausdorff distance** is defined as

$$\delta_h(P, Q) = \max \left\{ \max_{v \in V(P)} d(v, Q), \max_{w \in V(Q)} d(w, P) \right\} \quad (5.7)$$

with $d(v, P) = \min \{d(v, w) \mid w \in V(P)\}$.

The construction with two maximum operators is due to the fact that otherwise the measure would not be symmetric. Consider Figure 5.5a as an example where the two walks P and Q are shown by blue (P) and green (Q) edges. The gray dashed lines are existing edges in the graph. Computing $\max_{v \in V(P)} d(v, Q)$ yields a value of 1 because for all nodes of P , the minimal distance to any node of Q is its distance to node v which is 0 or 1 for all nodes of P . Vice versa, computing $\max_{w \in V(Q)} d(w, P)$ yields a value of 4 because the closest node of P to w is v with a distance of 4.

Properties of the Hausdorff distance Although this distance measure is used often, for example, to compare trajectories of moving objects (for example [JJS04]) and is actually a distance metric in the original version, it does have several unwanted properties. First, when applied to walks, it does not satisfy coincidence anymore (see Figure 5.3). Second, it is obviously very sensitive to out-

liers since only the maximal distance is considered. In the case of walks, the term outliers is not well-suited because walks are restricted by the graph structure. However, taking the maximum distance between the nodes of two walks, as the Hausdorff measure does, often does not capture our intuition regarding the distance between two walks. Two walks with a constant distance of k should be rated differently than two walks that are identical for most of their parts, but contain a few nodes that have a distance of k . One possibility to improve this behavior is to take the sum of the corresponding distances instead of the maximal distance. This would allow respecting the "average" distance of the walks instead of their maximal distance. Furthermore, we need to determine which nodes of P to compare to which nodes in Q .

5.2.4 A generic scheme for distance measures for sets and sequences of points in a metric space

In general, there are a lot of possibilities for measuring the distance between two sets or sequences of points in a metric space when the position of the points is considered for the computation. We can give a generic definition of a distance measure from which several well-known distance measures can then be derived. We will define this generic measure for sequences of points in a metric space such that it is also applicable for the next section, where we will consider walks as sequences. Distance measures derived from the generic distance measure that do not consider the order of the points in the sequence are thus distance measures for *sets* of points.

Let $A = (a(1), a(2), \dots, a(k))$ and $B = (b(1), b(2), \dots, b(l))$ be sequences of points in a metric space (V, d) , thus $a(i), b(j) \in V$ for all $i \in I(A)$ and all $j \in I(B)$, where we denote $I(A) = \{1, \dots, k\}$ and $I(B) = \{1, \dots, l\}$ as the index set of A and B , respectively. For computing a distance measure between A and B , we need to determine which point in A is compared to which point in B . Thus, a mapping relation $\mu \subseteq I(A) \times I(B)$ is needed that determines which distances between a point of A and a point of B is used in the measure. If $(i, j) \in \mu$, we say that point $A(i)$ is mapped onto point $B(j)$. Second, for a given mapping relation μ , we need to determine *how* these distances are used in the measure; hence, a cost function for a given mapping relation needs to be defined. The cost function can, for example, take the maximum distance of all mapped points as in the Hausdorff distance, or it can take the sum of the distances of all mapped points. Furthermore, it needs to be determined which of the possible mapping relations is used for computing the distance measure. We thus define the following generic scheme for distance measures for sequences of points in a metric space:

Definition 5.5: Generic scheme for distance measures for sequences of points in metric space

For $A = (a(1), a(2), \dots, a(k))$ and $B = (b(1), b(2), \dots, b(l))$ sequences of points in a metric space (V, d) , a generic position-based distance measure for A and B is given by

$$\delta(A, B) = N \cdot \mathcal{C}_{\mu \in \mathcal{H}_{A,B}} \{cost(\mu)\}$$

with

Mapping relations For a set of specified mapping properties, $\mathcal{H}_{A,B}$ denotes the set of all mapping relations $I(A) \times I(B)$ with those properties.

Cost function For a mapping relation $\mu \in \mathcal{H}$, $cost(\mu)$ yields the costs of it for the given sequences A and B .

Choice of mapping For a given cost function $cost$ and a set of mapping relations $\mathcal{H}_{A,B}$, \mathcal{C} chooses a $\mu \in \mathcal{H}_{A,B}$ with respect to $cost(\mu)$. Here, in most cases \mathcal{C} is set to min.

Normalization factor N is a normalization factor which can be chosen depending on the chosen μ and the given sequences A and B .

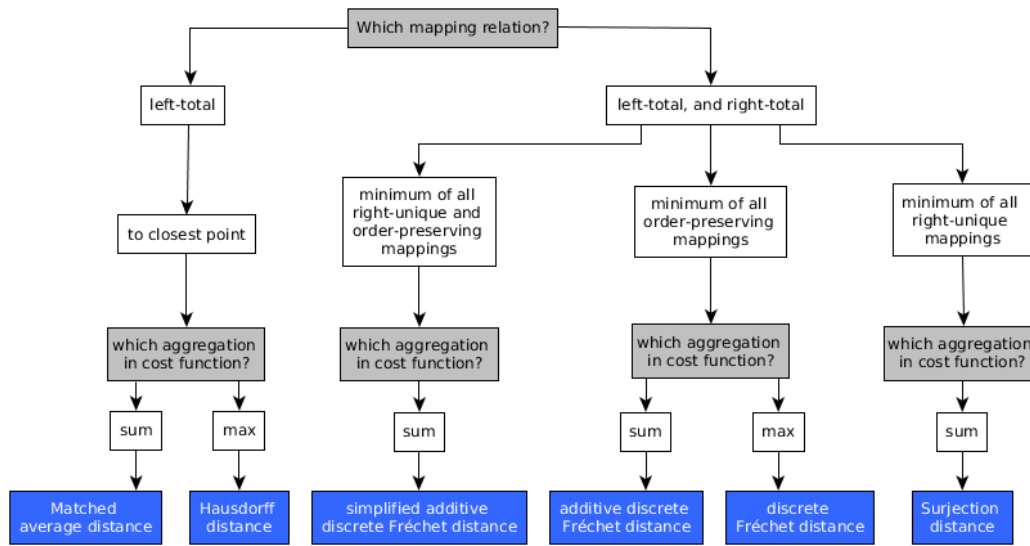


Figure 5.4 Which choices of mapping relations and of which aggregations in the cost function lead to which distance measures. For simplification, constructions to make the measures symmetric such as in the Hausdorff distance are omitted in this categorization.

We can require several properties of a mapping relation $\mu \subseteq I(A) \times I(B)$.

Definition 5.6: Properties of mapping relations

We say a relation μ is

left-total (lt) if for every $i \in A$, there exists a $j \in B$ with $(i, j) \in \mu$,

right-total (rt) if for every $j \in B$, there exists a $i \in A$ with $(i, j) \in \mu$,

left-unique (lu) if for all $i, i' \in A$ and all $j \in B$, it holds that $(i, j), (i', j) \in \mu \Rightarrow i = i'$ (injective),

right-unique (ru) if for all $i \in A$ and all $j, j' \in B$, it holds that $(i, j), (i, j') \in \mu \Rightarrow j = j'$ (functional),

order-preserving (op) if for every $(i, j), (i', j') \in \mu$, it holds that $i < i' \Leftrightarrow j \leq j'$ and $j < j' \Rightarrow i \leq i'$.

The relevant steps for instantiating a distance measure based on the given generic definition are thus: select the desired properties of the mapping relation (or select a specific mapping relation), select a cost function for the mapping relations, select the mapping (where mostly the one with the minimal costs is taken), and select a normalization factor. Figure 5.4 shows an overview of which selections lead to which distance measures, which will be introduced in the following.

The matched average distance

If a mapping relation such as the one implicitly contained in the Hausdorff distance is used, i.e. a left-total, right-unique relation where for each point in A , its closest point in B is mapped to it, and a cost function is selected that sums up the distances of the mapped points, we get a measure we call *matched average distance* [Boc15, BZ16]. In this case, the choice of the mapping is unique such that C does not need to be specified. For an unnormalized variant, we choose N to be 1; for a normalized variant, N is set to the length of the longer sequence. Our selection is aimed at assuring that each node of the longer walk is mapped to a node of the shorter walk.

Definition 5.7: Matched average distance for walks

For walks $P = (p_1, \dots, p_\ell)$ and $Q = (q_1, \dots, q_k)$, we define their **matched average distance** as

$$\delta_{mad}(P, Q) = \begin{cases} \sum_{i=1}^{\ell} d(p_i, Q) & \text{if } \ell > k \\ \sum_{i=1}^k d(q_i, P) & \text{if } \ell < k \\ \min\{\sum_{i=1}^{\ell} d(p_i, Q), \sum_{i=1}^k d(q_i, P)\} & \text{otherwise} \end{cases} \quad (5.8)$$

and the **normalized matched average distance** as

$$\delta_{mad}^N(P, Q) = \frac{\delta_{mad}(P, Q)}{\max\{|P|, |Q|\} + 1}. \quad (5.9)$$

Note that the order of the nodes in the walks is not taken into account, so this measure is for *sets* of points in the metric space (V, d) .

Properties of the matched average distance With this mapping relation, it can happen that there are nodes in the shorter walk that are not taken into account at all, which is not a desired property. See Figure 5.5a for an example: All nodes of the blue path are mapped onto node v of the green path while the position of all other nodes of the green path is not used in the computation. We hence need to require further properties of the mapping relation μ . Furthermore, it is not a distance metric because it does not satisfy coincidence and the triangle inequality.

Further distance measures for sets of points in metric space

There are several other distance measures for sets of points in a metric space, for example the surjection measure proposed by the philosopher Oddie for comparing theories to each other [Odd86]. This measure is gained when the cost-optimal mapping of all left-total, right-total and right-unique mapping relations is used in the introduced generic scheme, and the distances of all mapped points are summed up. Further measures that could be adapted for walks are the fair surjection distance (where the entries of the mapping relation need to be fairly distributed among all nodes of P and Q , see [EM97]); a measure called link distance where left-total and right-total mappings are considered [EM97]; and the matching distance proposed by Ramon and Bruynooghe [RB01].

5.2.5 A walk as a polygonal curve

It seems natural to combine the previous approaches and to define a distance or similarity measure that incorporates all three kinds of information contained in a walk: its elements, the order of its elements, and the position of its elements in the graph. A data structure containing all these kinds of information is a sequence of points in a metric space (in computational geometry, this is called a polygonal curve or polygonal chain). Polygonal curves are often the result of the discretization of a continuous curve in a metric space. Informally, a polygonal curve is a sequence of points in a metric space where two consecutive points are connected by a straight line.

Definition 5.8: Continuous and polygonal curves

A *curve* is a continuous mapping $f : [a, b] \rightarrow V$, where $a, b \in \mathbb{R}$ and $a < b$ and (V, d) is a metric space.

A *polygonal curve* in a metric space (V, d) is a mapping $C : [0, k] \rightarrow V$ (where $k \in \mathbb{N}$), such that for each $i \in \{0, 1, \dots, k-1\}$ and $\lambda \in [0, 1]$, it holds that

$$C(i + \lambda) = C(i) + \lambda(C(i+1) - C(i)).$$

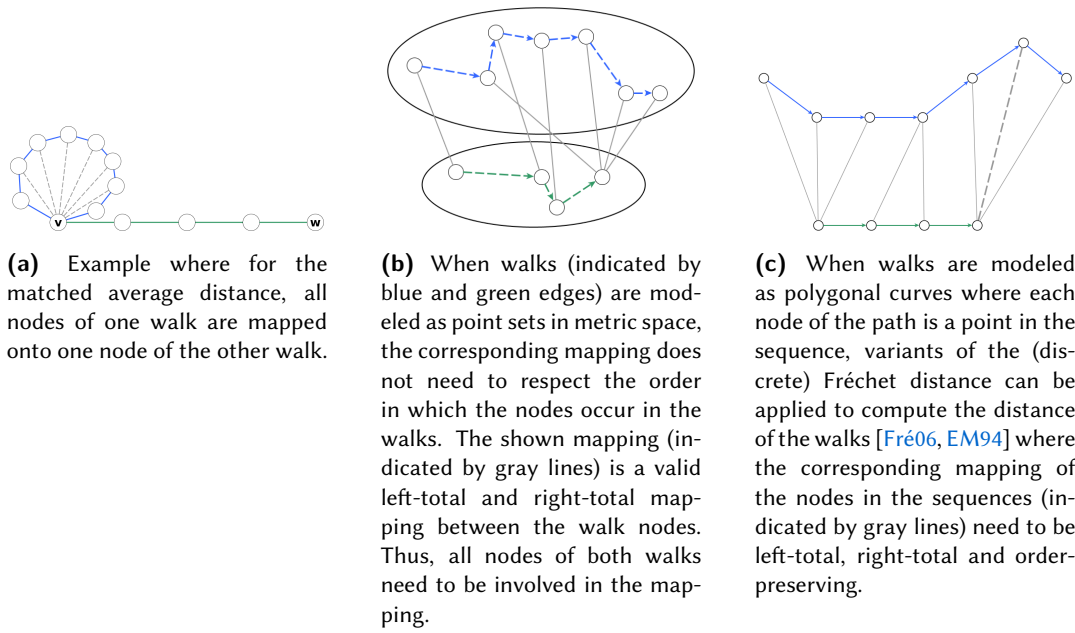


Figure 5.5 Modeling of walks as sets and sequences of points in a metric space and possible resulting distance measures.

A polygonal curve is hence uniquely determined by the sequence $(C(0), C(1), \dots, C(k))$.

There exist several distance measures for polygonal curves. A well-known one is the Fréchet distance [Fré06]. The Fréchet distance was originally proposed by Fréchet in 1906 (who, as a side note, also introduced the concept of metric spaces) as a distance measure for (continuous) curves in a metric space.

Definition 5.9: Fréchet distance for (continuous or polygonal) curves

Let $f : [a, b] \rightarrow V, g : [a', b'] \rightarrow V$ be curves in a metric space (V, d) , their Fréchet distance is defined as

$$\delta_F(f, g) = \inf_{\alpha, \beta} \max_{t \in [0, 1]} d(f(\alpha(t)), g(\beta(t))) \quad (5.10)$$

where α and β are arbitrary continuous nondecreasing functions from $[0, 1]$ to $[a, b]$ and to $[a', b']$, respectively.

A popular intuitive explanation for the distance measure is as follows: Assume a man walking with his dog on a leash. The man is following one curve, his dog another one. Their speed can vary, but they can only move forward and they need to go from the start to the end of their curve. The Fréchet distance between the two curves is then the minimal length of the leash that is sufficient to allow both to follow their routes. The man and the dog adapt their speed in such a way that the necessary length of the leash decreases. Transferring this example to the formal definition, the parameter t can be understood as time, then $f(\alpha(t))$ is the position of the man (or the dog) at time t . The distance between dog and man at time t is thus $d(f(\alpha(t)), g(\beta(t)))$. The Fréchet distance then uses the maximal distance between man and dog at any time point (maximizing over all time points $t \in [0, 1]$) when both keep track of their speed in the optimal way (taking the infimum over all possible variations of speed α and β). This definition can also be applied to polygonal curves as

a special case of continuous curves.

The discrete Fréchet distance

For the comparison of walks in graphs, a variant of the Fréchet distance is interesting, which is the *discrete Fréchet distance* δ_{dF} where only the points $C(0), C(1), \dots, C(k)$ are considered for the computation, but not the straight lines between these line endpoints. The discrete Fréchet distance is an approximation of the Fréchet distance and was introduced by Eiter and Mannila [EM94]. For two polygonal curves A and B , the illustrative example can be adapted as follows: The man and the dog are replaced by two frogs jumping from one stone ($A(i)$ and $B(i)$) to another ($A(i+1)$ and $B(i+1)$) [AAKS14]. As before, the frogs can choose the point of time when they jump to the next stone, but they need to start at $A(0)$ and $B(0)$, respectively, and reach $A(k)$ and $B(l)$ and are only allowed to jump to the next stone. The discrete Fréchet distance of A and B is then the minimal length of the leash connecting the frogs that is sufficient for taking their route. Note that the Fréchet distance of two polygonal curves and the discrete Fréchet distance of the same two polygonal curves are not necessarily equal. Eiter and Mannila [EM94] defined the discrete Fréchet distance for polygonal curves by introducing a mapping relation (they call it coupling) between the curves which we can adopt to walks in graphs:

Definition 5.10: Discrete Fréchet distance for walks

Given two walks P and Q in a graph G , let $\mathcal{G}_{P,Q}$ denote the set of all left-total, right-total and order-preserving mapping relations between $I(P)$ and $I(Q)$. For a mapping $\mu \in \mathcal{G}_{P,Q}$, its cost is then defined as the length of the longest link

$$\text{cost}_m(\mu) = \max_{(i,j) \in \mu} d(P(i), Q(j))$$

and the discrete Fréchet distance can then be defined as the cost of the cost-minimal mapping:

$$\delta_{dF}(P, Q) = \min_{\mu \in \mathcal{G}_{P,Q}} \text{cost}_m(\mu) \quad (5.11)$$

The mapping with the minimal costs is called the optimal mapping (and denoted by μ^*).

The discrete Fréchet distance incorporates the order of the contained points as well as their position. Note that there exist continuous and discrete versions of both: the curves (yielding continuous and polygonal curves) and the measure (yielding the continuous and the discrete Fréchet distance). Relevant for this work is the discrete Fréchet distance for polygonal curves. It fulfills all desired properties, even the triangle inequality [EM97].

Additionally, we will use an additive variant of the discrete Fréchet distance where the cost of a mapping μ is not defined by its *longest* link, but by the sum of its links:

Definition 5.11: Additive discrete Fréchet distance for walks

For two walks P and Q in a graph G , let $\mathcal{G}_{P,Q}$ denote the set of all left-total, right-total and order-preserving mapping relations between $I(P)$ and $I(Q)$. For a mapping $\mu \in \mathcal{G}_{P,Q}$, its cost is defined as the sum of the links. The discrete Fréchet distance can then be defined as the cost of the cost-minimal mapping:

$$\text{cost}_\Sigma(\mu) = \sum_{(i,j) \in \mu} d(P(i), Q(j))$$

With this additive cost function, we define for two walks P and Q , their **additive discrete**

Fréchet distance as

$$\delta_{adF}(P, Q) = \min_{\mu \in \mathcal{G}} \text{cost}_{\Sigma}(\mu) \quad (5.12)$$

and their **normalized^a additive discrete Fréchet distance as**

$$\delta_{adF}^N(P, Q) = \frac{1}{\max\{|P|, |Q|\} + 1} \min_{\mu \in \mathcal{G}_{P,Q}} \text{cost}_{\Sigma}(\mu). \quad (5.13)$$

^aNote that unweighted and simple graphs are assumed in this chapter.

Properties of the additive discrete Fréchet distance By changing the cost function from using the *longest* link to using the *sum* of the links, the distance measure δ_{adF} loses some properties of distance metrics; in particular, the triangle inequality is not fulfilled anymore, not even for walks of the same length: Figure 5.6 shows three walks in a graph for which the triangle inequality is violated:

$$\begin{aligned} \delta_{adF}(P, Q) &= 6 \\ \delta_{adF}(Q, R) &= 6 \\ \delta_{adF}(P, R) &= 14 \end{aligned}$$

and thus,

$$\delta_{adF}(P, Q) + \delta_{adF}(Q, R) < \delta_{adF}(P, R)$$

Figure 5.6b shows the reason why the triangle inequality does not hold: On the left, one of the cost-optimal mappings between P and Q is shown as well as one of the cost-optimal mappings between Q and R , on the right, a cost-optimal mapping between P and R is shown. Mapping P and R directly is more expensive than adding the costs of the single mappings because the cost of the edges $(A, 1)$ and $(5, V)$ are only counted once in the single mappings, but are counted three times in the direct mapping between P and R .

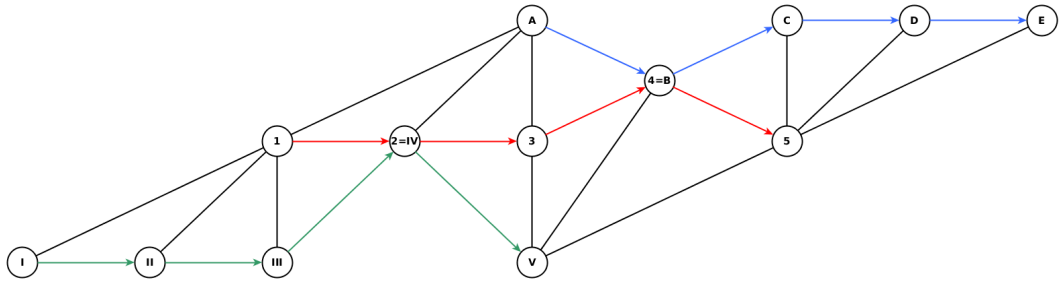
Eiter and Mannila propose an algorithm computing the discrete Fréchet distance with a runtime of $\mathcal{O}(kl)$ with a dynamic programming approach where k and l are the number of points in the sequence of the polygonal curves, Agarwal et al. [AAKS14] provide an algorithm computing the discrete Fréchet distance with a subquadratic runtime, with a runtime of $\mathcal{O}\left(\frac{kl \log(\log(l))}{\log(l)}\right)$ when $l \geq k$.

A simplified additive discrete Fréchet distance δ_{sadF}

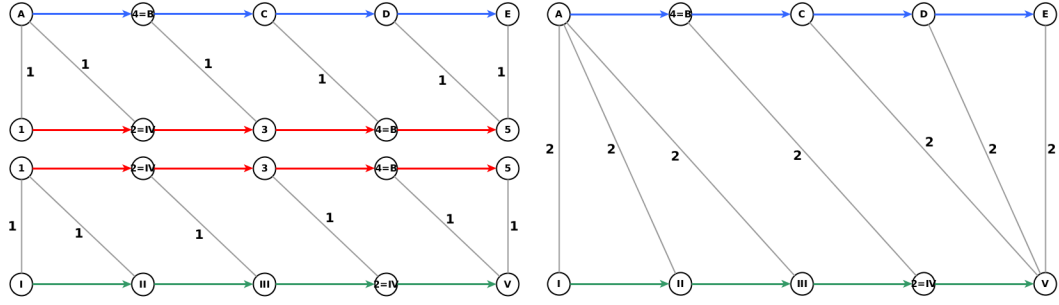
We furthermore consider a slightly different variant of δ_{adF} that is easier to compute than δ_{adF} . We call this variant *simplified additive discrete Fréchet distance* δ_{sadF} . In this variant, the requirements for the mapping relation are constrained such that only mapping *functions* (instead of relations) are allowed, i.e, the relation is required to be right-unique. Since for the existence of a left-total, right-total, and right-unique relation between $I(P)$ and $I(Q)$, it must hold that $|I(P)| \geq |I(Q)|$, we consider mapping functions from the longer walk to the shorter walk.

Definition 5.12: Simplified discrete additive Fréchet distance for walks

For two walks P and Q in a graph G , let $\mathcal{F}_{P,Q}$ denote the set of all left-total, right-total, right-unique and order-preserving mapping relations between $I(P)$ and $I(Q)$. For a map-



(a) Three walks in a graph; black edges are existing edges in the graph.



(b) Left: a cost-optimal mapping (denoted by gray lines) between P and Q , and a cost-optimal mapping between Q and R (with respect to the cost function $cost_{\Sigma}$). Right: a cost-optimal mapping between P and R (the edge labels denote the cost of including the edge in the mapping).

Figure 5.6 An example of three walks for which the triangle inequality for the additive discrete Fréchet distance δ_{adF} and δ_{adF}^N fails. It can be seen why the triangle inequality fails here: it is less expensive to add up the cost of the single mappings than directly map P and Q because when adding the costs of the single mappings, the cost of the edges $(A, 1)$ and $(5, V)$ are only counted once while for the mapping between P and Q , these edges need to be counted three times.

ping $\mu \in \mathcal{F}_{P,Q}$, its cost is defined as the sum of the links:

$$cost_{\Sigma}(\mu) = \sum_{(i,j) \in \mu} d(P(i), Q(j))$$

With this additive cost function, we define for two walks P and Q , their **simplified additive discrete Fréchet distance** as

$$\delta_{sadF}(P, Q) = \begin{cases} \min_{g \in \mathcal{F}_{P,Q}} cost_{\Sigma}(g) & \text{if } |P| \geq |Q| \\ \min_{g \in \mathcal{F}_{Q,P}} cost_{\Sigma}(g) & \text{otherwise} \end{cases} \quad (5.14)$$

and their **normalized simplified additive discrete Fréchet distance** as

$$\delta_{sadF}^N(P, Q) = \frac{1}{\max\{|P|, |Q|\} + 1} \delta_{sadF}(P, Q). \quad (5.15)$$

A simple observation leads to an algorithm computing δ_{sadF} which has the same scheme as the algorithm proposed by Eiter and Mannila for the discrete Fréchet distance [EM94]. The algorithm (Algorithm A.2) and a proof of correctness are provided in the Appendix in Section A.3.

5.3 Clustering of trajectories

The previous sections introduced a variety of similarity measures and distance measures for walks in a graph. We introduced these measures with the aim of using them for clustering empirically observed walks into meaningful groups. Thus, the introduced measures were tested in order to determine whether they are suited for finding meaningful groups of walks in a graph to enable answering the following question:

Which of the introduced similarity and distance measures is best suited for finding meaningful clusterings of trajectories in a graph?

For this purpose, a dataset with a ground truth of meaningful groups is needed so that the quality of the clustering computed by the distance and similarity measures can be evaluated by the correct clustering given by the ground truth. We used a dataset containing state spaces of a single-player board game and attempts of human players to solve the game which are walks through the state space. For each pair of walks in the same state space, all distance and similarity measures introduced in the previous sections of this chapter were computed. Then, a standard clustering procedure was applied to each distance and similarity measure. The advantage of this dataset is the availability of a very simple ground truth: A clustering based on a similarity or distance measure should at least be able to distinguish solving and non-solving attempts. We thus evaluated the quality of the computed clusters according to this ground truth.

5.3.1 Clustering procedures

In general, the task of clustering is as follows: Given a set of elements $S = \{s_1, s_2, \dots, s_m\}$, find a grouping of the elements $\Gamma = \{\gamma_1, \gamma_2, \dots, \gamma_l\}$ (also called clustering) where $\gamma_i \subseteq S$ for all $i \in \{1, \dots, l\}$ and $\bigcup_{\gamma_i \in \Gamma} \gamma_i = S$. A clustering is non-overlapping (also sometimes called crisp) if the single clusters of Γ are disjoint, i.e., $\gamma_i \cap \gamma_j = \emptyset$ for all $i \neq j$. Then, the grouping Γ is also a partition. If elements of S are allowed to be in more than one $\gamma_i \in \Gamma$, the clustering is called overlapping (or sometimes fuzzy).

There are many approaches for finding such clusterings. They all have in common that they are given a set of elements and return a grouping of these. Since in this work, we aim at evaluating similarity and distance measures, relevant clustering approaches are those based on the distance or similarity between the elements of S . There also exist other approaches, for example based on features computed for each element. Clustering approaches based on similarity or distance measures require a distance or similarity measure defined on $S \times S$ and are designed to find a clustering such that elements that are closer (or more similar) to each other belong to the same cluster while elements that are less similar to each other belong to different classes.

Probably the best-known classes of distance-based methods are hierarchical and centroid-based clustering approaches. We will use a hierarchical approach, which we will explain briefly in the following.

Hierarchical clustering approaches The idea of hierarchical clustering approaches is that a tree of the elements is built: Each element is a leaf of the tree and “cutting the tree” at a certain height yields a clustering of elements. Cutting the tree disconnects the tree and the resulting components represent the computed clusters.

For the computation of the tree (also called a dendrogram), approaches proceed either in an agglomerative or in a divisive way. In divisive approaches, the tree is built top-down, i.e., starting

from the root where all elements are in one cluster, and iteratively dividing the existing clusters into smaller ones. In agglomerative approaches, the tree is built bottom-up, thus, in the beginning, each element is in its own cluster (contained in the tree leaves), which are merged into larger clusters (subtrees) step by step. Thus, each step of the procedure (and hence, each level of the tree) represents a valid grouping of the elements. When the tree is built, it needs to be decided which of the possible clusterings is used as the final clustering.

Taking an agglomerative approach as an example, it proceeds as follows:

- Given: a set of elements $S = \{s_1, \dots, s_m\}$, a distance measure² $\delta : S \times S$, and a distance measure D for clusters (explained below).
- As initialization, each element $s \in S$ is assigned to its own cluster.
- In each step, the distance D between each pair of clusters is computed (as explained below), and the two clusters with the smallest distance are merged. This is repeated until all elements are contained in the same cluster.

For measuring the distance of two clusters $\gamma_1, \gamma_2 \subseteq S$, there are several measures. The best-known ones are:

Single linkage For two clusters γ_1 and γ_2 , their distance is computed by considering the elements of γ_1 and γ_2 that are closest to each other with respect to the distance measure. Thus, we get

$$D_{sl}(\gamma_1, \gamma_2) = \min_{s_i \in \gamma_1, s_j \in \gamma_2} d(s_i, s_j)$$

Complete linkage The distance of two clusters is based on the elements of γ_1 and γ_2 that are furthest apart with respect to the distance measure d and thus,

$$D_{cl}(\gamma_1, \gamma_2) = \max_{s_i \in \gamma_1, s_j \in \gamma_2} d(s_i, s_j)$$

Average linkage (UPGMA) Here, the distance of two clusters is based on the average distance between the elements of γ_1 and the elements of γ_2 :

$$D_{av}(\gamma_1, \gamma_2) = \frac{1}{|\gamma_1| \cdot |\gamma_2|} \sum_{s_i \in \gamma_1} \sum_{s_j \in \gamma_2} d(s_i, s_j)$$

Hierarchical clustering approaches have the advantage that the final number of clusters does not need to be known in advance. Furthermore, the structure of the built tree can give insights about the structure of the data. On the other hand, in order to get a clustering, it needs to be determined which clustering to choose. A further property of hierarchical clustering procedures which is an advantage and a drawback at the same time, is the fact that elements do not change their cluster affiliation during the clustering procedure: Elements placed into the same cluster at some step of the procedure will stay in the same cluster. There are cases where it might be beneficial to reassign elements to clusters in a later stage of the procedure; however, this is not possible in hierarchical clustering approaches. At the same time, this property is also an advantage of the approach: Since the decision about an element's cluster affiliation is irreversible, the number of possibilities that need to be considered is reduced significantly. This has an impact on the computational complexity of the procedure.

5.3.2 Dataset used

As a dataset for evaluating the introduced similarity and distance measures for walks in graphs, we used a dataset containing twenty different configurations of the board game Rush Hour as

²The procedure can be easily adapted when a similarity measure instead of a distance measure is given

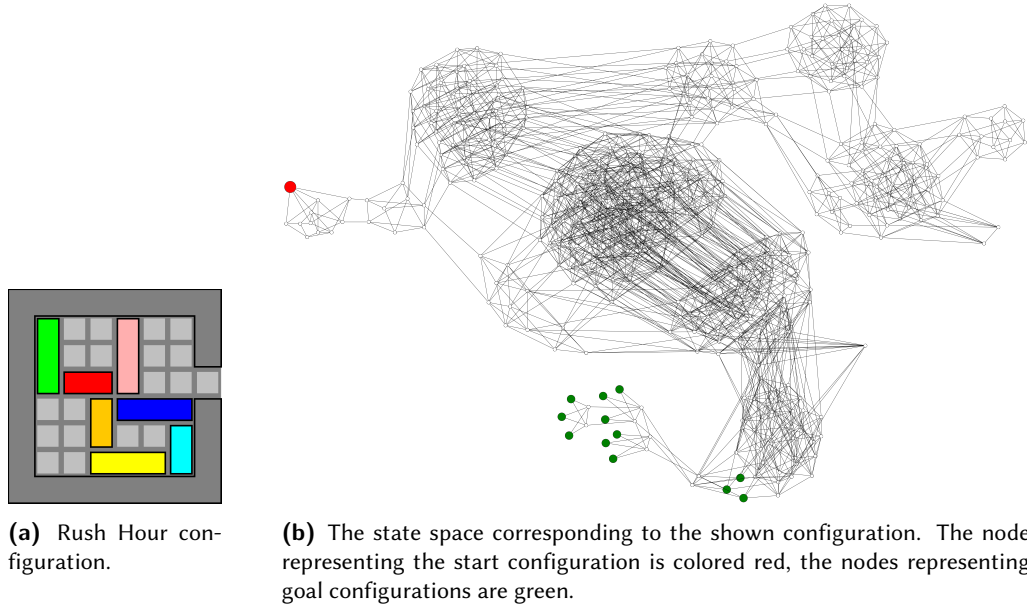


Figure 5.7 A Rush Hour game configuration and its resulting state space. The goal is to arrange the cars in such a way that the red car can leave through the exit on the right side.

described in Chapter 3. Unlike the data used in Chapter 3, in this chapter, the state spaces and the logs of twenty game configurations instead of only three are used; furthermore, here, all solution attempts—not only the successful ones—are included in the analysis.

Thus, a set of twenty game configurations with almost 14000 walks in total was used as the evaluation dataset, of which roughly 6000 walks were solving walks. Table 5.2 shows for each of the selected games the size and order of its state space as well as how many solution attempts were available for it. For a game configuration c , let $G_c = (V_c, E_c)$ denote the corresponding state space, and \mathcal{P}_c denote the set of available walks in G_c .

5.3.3 Clustering procedure

We performed the following procedure (see also Figure 5.8):

- For each starting configuration c , and each pair of walks in \mathcal{P}_c , all distance and similarity measures described in Section 5.2 were computed (normalized and unnormalized). This yields a matrix of size $|\mathcal{P}_c| \times |\mathcal{P}_c|$ for each configuration and each measure.
- The unnormalized measure values were scaled to the interval $[0, 1]$ using the following transformation: Let A denote the matrix containing the measure values where a_{ij} is an entry of A , then all entries are substituted by

$$a'_{ij} = \frac{a_{ij} - \min_{k,l} a_{kl}}{\max_{k,l} a_{kl} - \min_{k,l} a_{kl}}$$

for all i, j . This yields a matrix containing values between 0 and 1.

- The similarity measures were then converted to distance measures by inverting their scale: If A again denotes the matrix with the similarity measures (from the interval $[0, 1]$ due to normalization or the previous scaling), all entries of A are substituted by

$$a'_{ij} = 1 - a_{ij}$$

for all i, j . Thus, all matrices contain values between 0 and 1 where 0 means high similari-

Table 5.2 Overview of the dataset used. For each configuration, the size of the associated problem space is shown ($|V|$ and $|E|$ denote the number of nodes and edges of the problem space, $|V^{\mathcal{P}}|$ denotes the number of nodes occurring in at least one of the walks, % is the percentage of nodes used). Optimal denotes the length of the optimal solution path. The remaining columns contain information on the number of solution attempts contained in the dataset for each configuration and the number of solving and non-solving attempts.

	State spaces			$ E $	optimal	total	Number of walks	
	$ V $	$ V^{\mathcal{P}} $	%				solving	non-solving
Game 19	1169	153	13.01	8620	31	662	213	449
Game 64	2952	354	12.00	21017	5	2934	2592	342
Game 121	4405	263	5.97	33302	47	270	39	231
Game 202	4635	171	3.69	38176	41	359	89	270
Game 246	3003	323	10.76	22418	33	552	158	394
Game 260	3095	203	6.56	24919	48	247	54	193
Game 326	3493	175	5.01	27529	50	290	48	242
Game 357	4426	99	2.24	37649	42	205	58	147
Game 393	4533	244	5.38	30587	49	175	53	122
Game 441	4533	238	5.25	30587	49	178	59	119
Game 578	2853	257	9.01	24732	31	904	230	674
Game 579	4573	189	4.13	35232	30	511	150	361
Game 674	6090	128	2.10	53537	44	306	90	216
Game 692	887	126	14.21	5226	46	404	89	315
Game 722	2241	144	6.43	14517	48	156	47	109
Game 723	830	181	21.81	7978	13	2704	1472	1232
Game 765	1327	182	13.72	10143	30	462	109	353
Game 820	7235	204	2.82	63551	41	212	44	168
Game 841	1050	128	12.19	5957	45	203	65	138
Game 906	864	226	26.16	6934	24	2013	520	1493

ty/small distance and 1 means low similarity/large distance.

- Each matrix was input into a hierarchical clustering procedure with complete linkage. Using a hierarchical clustering procedure allows considering the complete clustering procedure between 1 big cluster containing all walks and $|\mathcal{P}_c|$ clusters containing one walk each. This is convenient since it was not clear what the “correct” number of clusters was for this dataset. We used the function `hc1ust` from the standard R library with the option of complete linkage (other linkage options were tested, but the results are quantitatively similar). This yielded a clustering object containing the cluster association of each walk in each step of the agglomerative hierarchical clustering procedure, for each game and for each distance and similarity measure.
- For a comparison of the quality of the clustering results (how we measured the quality of the clusterings will be explained below), we used two baseline clusterings: As an upper bound, we used a ground-truth-based distance measure which is 0 if two walks are both solving or both non-solving, and 1 otherwise. A matrix containing these values for each game was also input into the clustering procedure. As a lower bound, for each game c , 500 matrices of size $|\mathcal{P}_c| \times |\mathcal{P}_c|$ are filled with random numbers, drawn uniformly at random from the interval $[0, 1]$. Each of those random matrices was input into the clustering procedure, and the minimum, maximum and average quality of the 500 clustering results is computed.

5.3.4 Evaluating the quality of the clusterings

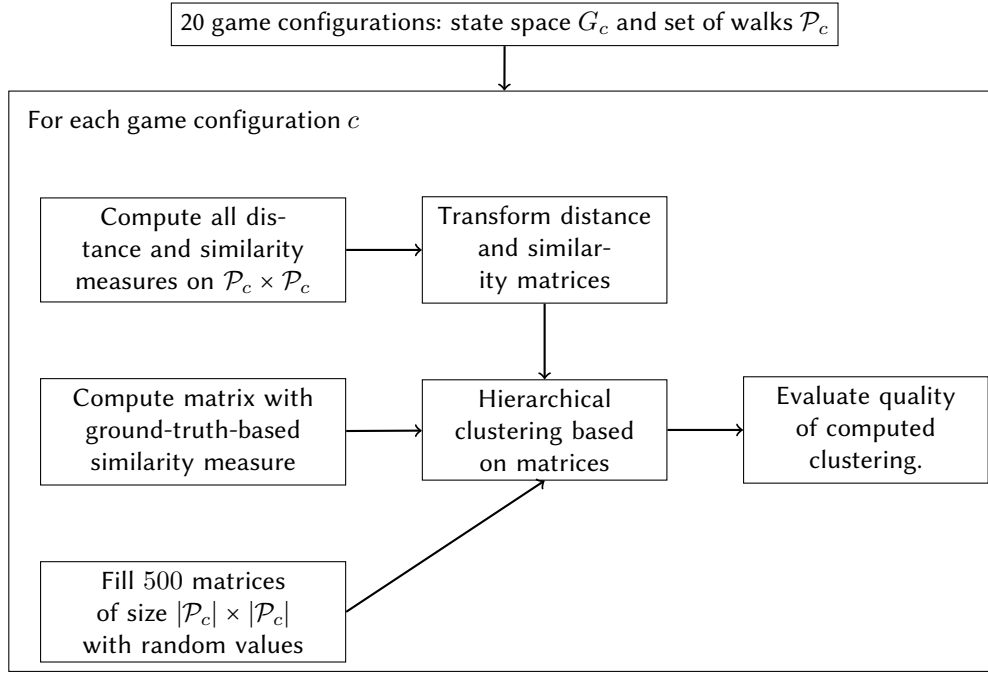


Figure 5.8 Procedure for evaluating the similarity and distance measures introduced in Section 5.2.

As a ground truth for the described dataset, the property of walks being solving or non-solving was used. The underlying assumption is that a similarity or distance measure should at least be able to distinguish between these types of walks. It is clear that this is not a perfect ground truth: Two solving walks might be very different, and two almost identical walks where one ends in a goal configuration and the other ends one step before (and is thus non-solving) should be considered as similar. Nevertheless, we assumed that the majority of the walks do not show these extreme cases, and thus, a solving and a non-solving walk should be structurally different. Furthermore, since the complete hierarchical clustering procedure was considered, this accounted for the possibility of several clusters of solving or non-solving walks.

For a clustering of a set of walks, we considered the purity of the clusters regarding the property of being solving or non-solving. For a walk $p \in \mathcal{P}_c$, let $q(p) \in \{0, 1\}$ denote this binary attribute where $q(p) = 1$ if p is solving, and $q(p) = 0$ if p is non-solving. For a given clustering $\Gamma = (\gamma_1, \dots, \gamma_k)$ of \mathcal{P}_c , a cluster $\gamma \in \Gamma$ is called *pure* if all walks in γ are either solving or non-solving. We could use the number of pure clusters as an evaluation criterion; however, requiring that all clusters should be pure is a very strict criterion. This is why we consider the *purity* of the clusters instead which tells how many of the walks contained in a cluster have the same property [MRS08]:

Definition 5.13: Purity of a cluster

Thus, for a cluster $\gamma \in \Gamma$, we define its purity as

$$\text{purity}(\gamma) = \frac{1}{|\gamma|} \max \left\{ \sum_{p \in \gamma} q(p), |\gamma| - \sum_{p \in \gamma} q(p) \right\},$$

i.e., the maximum of the two fractions of walks in γ which are solving or non-solving.

Note that $\text{purity}(\gamma) \geq 0.5$ always holds. For a clustering $\Gamma = (\gamma_1, \dots, \gamma_k)$, we consider two ways

of aggregating the purities of the single clusters. The *unweighted average purity* takes the mean purity of the single clusters:

Definition 5.14: Unweighted average purity of a clustering

For a clustering $\Gamma = (\gamma_1, \dots, \gamma_k)$ of \mathcal{P}_c , the unweighted average purity is defined as

$$purity(\Gamma) = \frac{1}{|\Gamma|} \sum_{i=1}^k purity(\gamma_i)$$

An unweighted average purity of a clustering has the effect that it is higher if Γ contains many small clusters, or even singletons, i.e., clusters containing only one element, because each singleton cluster contributes a value of 1 to the unweighted average purity. This is, however, not a good behavior of a quality measure for a clustering. We therefore consider a *weighted average purity* where the purity of each cluster of Γ is included proportionally to its size.

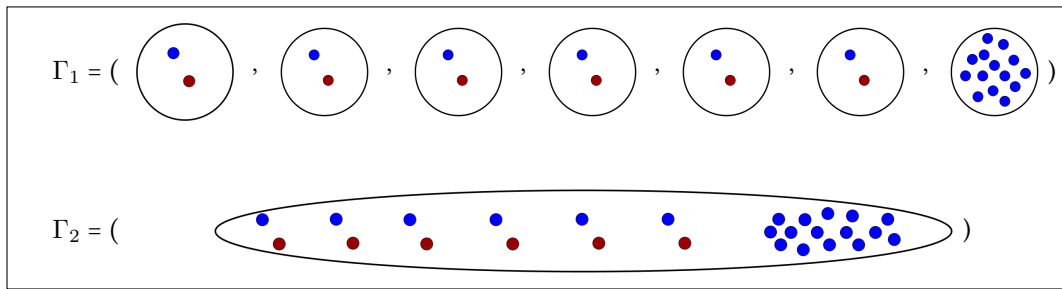
Definition 5.15: Weighted average purity

For a clustering $\Gamma = (\gamma_1, \dots, \gamma_k)$ of \mathcal{P}_c , the weighted average purity is defined as

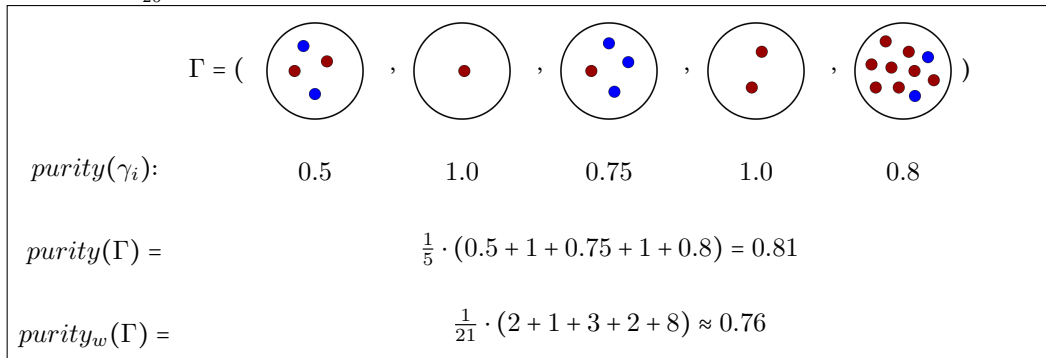
$$\begin{aligned} purity_w(\Gamma) &= \frac{1}{|\mathcal{P}_c|} \sum_{i=1}^k |\gamma_i| \cdot purity(\gamma_i) \\ &= \frac{1}{|\mathcal{P}_c|} \sum_{i=1}^k \max_{p \in \gamma_i} \sum_{p \in \gamma_i} q(p), |\gamma_i| - \sum_{p \in \gamma_i} q(p) \end{aligned}$$

Note that the weighted and unweighted average purity does not necessarily increase when clusters are split up. A good clustering procedure should do that, but there exist sequences of clusterings such that a clustering with a smaller number of clusters has lower (or equal) average purity as a clustering with a higher number of clusters. Figure 5.9a shows such a case.

The unweighted and weighted average purity is a quality measure for a given clustering Γ . In our procedure, we considered the complete hierarchical clustering procedure which gives a sequence of clusterings. In order to compare the quality of one clustering sequence to that of another one, we use the following idea: A good sequence of clusterings should reach high (weighted or unweighted) average purity with a small number of clusters and increase its average purity with an increasing number of clusters. In order to evaluate the quality of a sequence of clusterings, we adapted the idea of computing the area under the curve for the (weighted or unweighted) average purity line plotted against the number of clusters in the clustering (see Figure 5.10). In order to account for the fact that the walks of different game configurations have different ratios of solving to non-solving walks, we considered the area between the average purity line and the horizontal line with this ratio (the dashed line in Figure 5.10). Let AUC denote the size of this area (the red area in Figure 5.10). Let $q(\mathcal{P}_c)$ denote the maximum of the ratio of solving walks to all walks and the ratio of non-solving walks to all walks. A perfect clustering sequence has an average purity of $q(\mathcal{P}_c)$ for the clustering with one cluster, and an average purity of 1 for all other clusterings in the sequence. We therefore normalized AUC by the size of the area between the purity of a perfect clustering sequence and the q -line (the gray area in Figure 5.10) and got the *Relative AUC*.



(a) Example for the case where merging the clusters will yield a higher unweighted average purity and an equal weighted purity: Clustering Γ_1 yields an unweighted average purity of $\text{purity}(\Gamma_1) = \frac{1}{7}(6 \cdot 0.5 + 1.0) \approx 0.57$ while merging all clusters of Γ_1 yields the new clustering Γ_2 with an unweighted average purity of $\text{purity}(\Gamma_2) = \frac{20}{26} \approx 0.77$. Here, the weighted average purity is equal for Γ_1 and Γ_2 .



(b) Example for computing the unweighted and weighted average purity.

Figure 5.9 Illustrating the introduced measures weighted and unweighted average purity of a clustering.

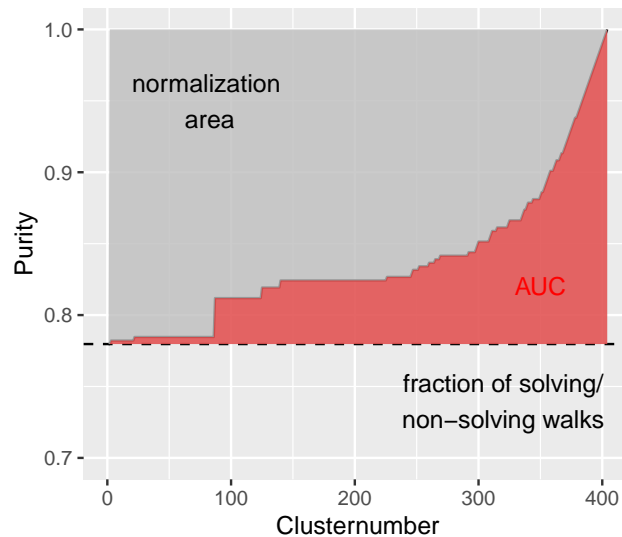


Figure 5.10 Example for computing the relative AUC: The average purity of a sequence of clusterings is plotted against the number of clusters. The relevant area AUC is the area between the purity line and the dashed line which is the maximum of the fractions of solving or non-solving walks in \mathcal{P}_c . In order to compute the relative AUC, the size of the red area is normalized by the size of the gray area.

5.3.5 Clustering results

This section presents the results of the clustering procedure, as described in the previous paragraphs.

How does the average purity evolve in the hierarchical clustering procedure?

Figure 5.11 shows the results for three example game configurations: For each game configuration (and its available game logs) and each similarity and distance measure, the hierarchical clustering procedure was performed. For each intermediate clustering in the clustering procedure, its unweighted and weighted average purity was computed. Figure 5.11 shows the average purity for each number of clusters. For the clusterings from random matrices, the average purity was computed for each clustering; then, the average, minimum and maximum of the average purity for each number of clusters was computed over the $N = 500$ iterations. In Figure 5.11, the mean value of the (weighted or unweighted) average purity of the random clustering is shown by a line, while the minimum and maximum are indicated by a gray area.

It is obvious that *any* clustering will reach an average purity of 100 % at some point of the clustering procedure, the latest when each element is in its own cluster. A good clustering, however, reaches a high average purity value with a small number of clusters.

It can be seen that the ground-truth-based clustering reached a (weighted and unweighted) average purity of 100 % with two clusters. Since existing clusters were not mixed during the procedure, but only merged, the purity value of 100 % was kept during the whole clustering procedure. For the clustering based on random values, it can be seen that the (weighted and unweighted) average purity increased approximately linearly with increasing numbers of clusters.

With these two baselines—the ground-truth-based clustering and the random clustering—we can evaluate the average purity lines of the clusterings based on the introduced similarity and distance measures. For almost all measures, the corresponding average purity lines are between the baselines. There are three measures—unnormalized node set similarity, unnormalized LCS similarity, and unnormalized edge set similarity—for which the clusterings have a considerably worse weighted average purity than the random clustering. Their purity values are very close to $q(\mathcal{P}_c)$ for the greatest part of the procedure, until they increase from a certain number of clusters on and reach 100 % with the maximal number of clusters. When comparing the unweighted average purity lines of these three measures, it can be seen that they have a local minimum at the same point where the weighted average purity starts to increase. This is due to the fact that the hierarchical clustering procedure for these three measures produces clusterings with a high number of singleton clusters, i.e., clusters with only one element. The increasing number of singletons, each with a purity of 100 %, leads to an increase of the unweighted average purity (as can be seen in Figure 5.11 on the left), but not to an increase of the weighted average purity. Then the effect described in Figure 5.9a occurs: Splitting up clusters does not increase the unweighted average purity, but leads to a decrease. This does not affect the weighted average purity. At some point, however, the clusters need to be split up further with an increasing number of clusters which leads to an increase of both weighted and unweighted average purity. This behavior was observed not only for the selected three configurations, but also for almost all other configurations.

However, such a behavior was only noted for the purely set- and order-based measures. The clusterings based on position-based measures show a higher weighted and unweighted average purity and are located between the average purities of the baseline clusterings. Especially the clusterings based on the three normalized variants of the measures incorporating all three kinds of information contained in a walk, i.e., normalized Fréchet distance, normalized additive discrete Fréchet distance, and normalized simplified additive discrete Fréchet distance, show an average purity that

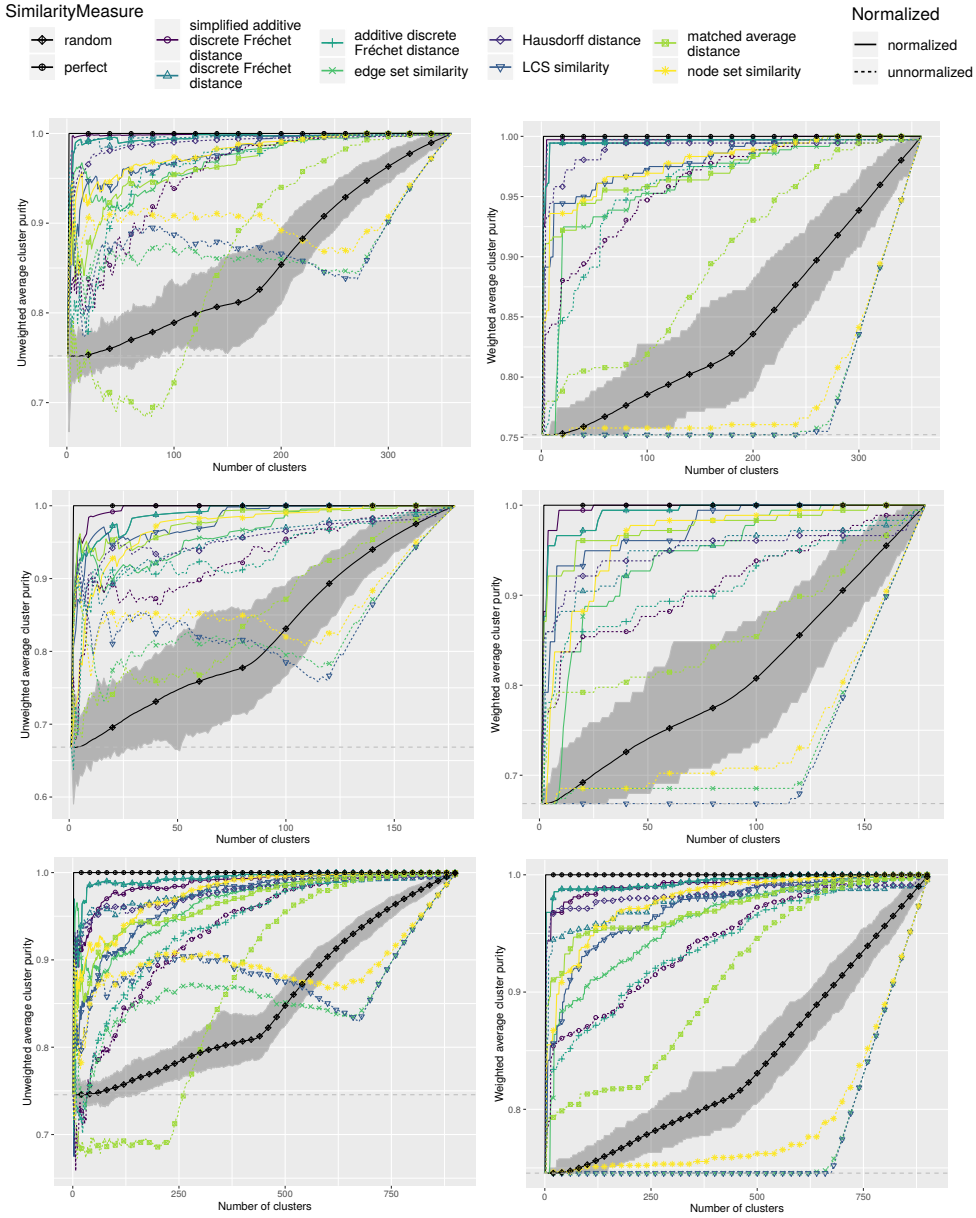


Figure 5.11 Unweighted average purity (left column) and weighted average purity (right column) of the clustering results for some example configurations, i.e., games 202, 441, and 578. The plots show the average purity during the complete hierarchical clustering procedure: The x -axis shows the number of clusters in the corresponding clustering, the y -axis its average purity. The horizontal gray dashed line indicates the $q(\mathcal{P}_c)$, i.e., the fraction of solving or non-solving walks for the corresponding game configuration. The gray shaded areas show the minimal and maximal average purity over all $N = 500$ simulations of random clustering.

is very close to the ground-truth-based clustering, and considerably better than the random clustering. Interestingly, for almost all measures, the normalized variants perform considerably better than the unnormalized variants. Even the normalized variants of the three measures with an average purity smaller than that for random clustering, node set similarity, edge set similarity, and LCS similarity, yielded clusterings with a weighted average purity above 90% with a reasonably small number of clusters for the shown game configurations. For all three shown configurations (and also for almost all other configurations), the clustering based on the unnormalized matched average distance is constantly and considerably lower than for the other measures, but higher than for random clustering.

Since it is desirable to achieve a high average purity with a small number of clusters, we considered the weighted average purity for a fixed (small) number of clusters and compared it for the different similarity and distance measures. Table 5.3 shows these numbers for x clusters with $x \in \{2, 5, 10, 20\}$ for three example game configurations. For each game and each x , the highest purity value is highlighted in the table. It can be seen that for such a small number of clusters, the distance measures incorporating a walk's elements, their order and their position in the graph, clearly yielded better clustering results than the measures incorporating only one of these features of a walk. When considering the results of all game configurations of the dataset and the clustering in terms of which measure yielded the highest weighted purity with only two clusters, we found that in all configurations where one measure clearly yielded a higher average purity, it was either the discrete Fréchet distance δ_{dF} (in five cases), the simplified additive discrete Fréchet distance δ_{sadF} (also in five cases), or the discrete additive Fréchet distance δ_{adF} or Hausdorff distance (in three cases each).

Relative AUC of average purity

In the previous paragraphs, the average purity of the clustering of single game configurations was described. In order to show that these findings are not just artifacts of single games, but are valid for the complete dataset, we computed the relative AUC of the average purity for each clustering, as described in Section 5.3.4, i.e., the area under the purity line normalized by the area under the “perfect” purity line. Figure 5.12 shows the relative AUC for each similarity and distance measure, aggregated over all game configurations of the dataset. It can be seen that the findings are similar to those for the single game configurations when aggregating over the data of all game configurations:

- The three distance measures incorporating the elements of a walk, their order *and* their position in the graph, yielded clusterings for which the relative AUC is close to the “perfect” clustering. The unnormalized variants of these three measures still had satisfactory results, but these were considerably worse than for the normalized variants. Furthermore, there is one game for which these three measures did not yield a good clustering result.
- The unnormalized variants of node set similarity, edge set similarity, and LCS similarity yielded clusterings with an aggregated relative AUC smaller than that for random clustering. Their normalized variants, though, performed considerably better.
- The relative AUC of the unnormalized matched average distance was considerably smaller than that for most other measures, but still better than the relative AUC of random clustering.

Does the clustering simply distinguish long and short walks?

In order to be a solving walk, a walk needs to have a minimum length, i.e., the length of the optimal solution. Non-solving walks do not have this length requirement. It is therefore possible that the clustering procedure only distinguishes between longer walks and shorter walks. Figure 5.13 shows the distribution of walk lengths in the single clusters for one specific game, when the clustering with two clusters is considered. For each distance and similarity, the lengths of the walks contained in cluster 1 and cluster 2 are shown.

Table 5.3 Weighted average purities of the clusterings for each of the introduced distance and similarity measures for a fixed number of clusters and a few selected game configurations. For each measure, the quality of the clusterings resulting from the unnormalized and the normalized distance/similarity measure are shown. p_x for $x \in \{2, 5, 10, 20\}$ denotes the weighted average purity when choosing the clustering with x clusters. $q(\mathcal{P}_c)$ denotes the fraction of solving or non-solving walks of all walks for the configuration. For each configuration and each $x \in \{2, 5, 10, 20\}$, the highest p_x for a similarity or distance measure is highlighted in green. The values of the ground-truth-based clusterings are not highlighted although they are always the highest.

Game	Measure	$q(\mathcal{P}_c)$	Normalized				Unnormalized			
			p_2	p_5	p_{10}	p_{20}	p_2	p_5	p_{10}	p_{20}
Game 19	δ_{sadF}	0.68	0.84	0.85	0.85	0.92	0.83	0.83	0.83	0.84
Game 19	δ_{dF}	0.68	0.84	0.90	0.91	0.96	0.85	0.86	0.89	0.89
Game 19	δ_{adF}	0.68	0.84	0.90	0.91	0.96	0.84	0.84	0.84	0.85
Game 19	s_{ess}	0.68	0.68	0.68	0.68	0.79	0.68	0.68	0.69	0.69
Game 19	δ_h	0.68	0.84	0.84	0.88	0.91				
Game 19	s_{lcs}	0.68	0.84	0.84	0.84	0.84	0.68	0.68	0.68	0.68
Game 19	δ_{mad}	0.68	0.82	0.84	0.85	0.89	0.68	0.75	0.81	0.81
Game 19	s_{nss}	0.68	0.74	0.79	0.79	0.84	0.68	0.68	0.76	0.76
Game 19	random	0.68	0.68	0.68	0.68	0.68				
Game 19	ground truth	0.68	1.00	1.00	1.00	1.00				
Game 357	δ_{sadF}	0.72	0.86	0.95	0.95	0.98	0.87	0.89	0.89	0.90
Game 357	δ_{dF}	0.72	0.94	0.99	1.00	1.00	0.96	0.96	0.97	0.97
Game 357	δ_{adF}	0.72	0.94	0.99	1.00	1.00	0.72	0.73	0.86	0.89
Game 357	s_{ess}	0.72	0.72	0.72	0.83	0.89	0.72	0.72	0.72	0.72
Game 357	δ_h	0.72	0.82	0.82	0.95	0.96				
Game 357	s_{lcs}	0.72	0.76	0.80	0.86	0.89	0.72	0.72	0.72	0.72
Game 357	δ_{mad}	0.72	0.84	0.85	0.90	0.90	0.76	0.79	0.80	0.81
Game 357	s_{nss}	0.72	0.72	0.85	0.85	0.86	0.72	0.72	0.72	0.72
Game 357	random	0.72	0.72	0.72	0.72	0.73				
Game 357	ground truth	0.72	1.00	1.00	1.00	1.00				
Game 723	δ_{sadF}	0.54	0.99	0.99	0.99	0.99	0.54	0.62	0.66	0.89
Game 723	δ_{dF}	0.54	1.00	1.00	1.00	1.00	0.93	0.96	0.97	0.97
Game 723	δ_{adF}	0.54	1.00	1.00	1.00	1.00	0.54	0.54	0.56	0.65
Game 723	s_{ess}	0.54	0.55	0.55	0.59	0.62	0.54	0.54	0.55	0.55
Game 723	δ_h	0.54	0.54	0.88	0.91	0.92				
Game 723	s_{lcs}	0.54	0.54	0.71	0.84	0.87	0.55	0.55	0.55	0.56
Game 723	δ_{mad}	0.54	0.88	0.95	0.95	0.95	0.54	0.84	0.84	0.84
Game 723	s_{nss}	0.54	0.62	0.86	0.90	0.94	0.55	0.55	0.55	0.57
Game 723	random	0.54	0.54	0.54	0.55	0.55				
Game 723	ground truth	0.54	1.00	1.00	1.00	1.00				

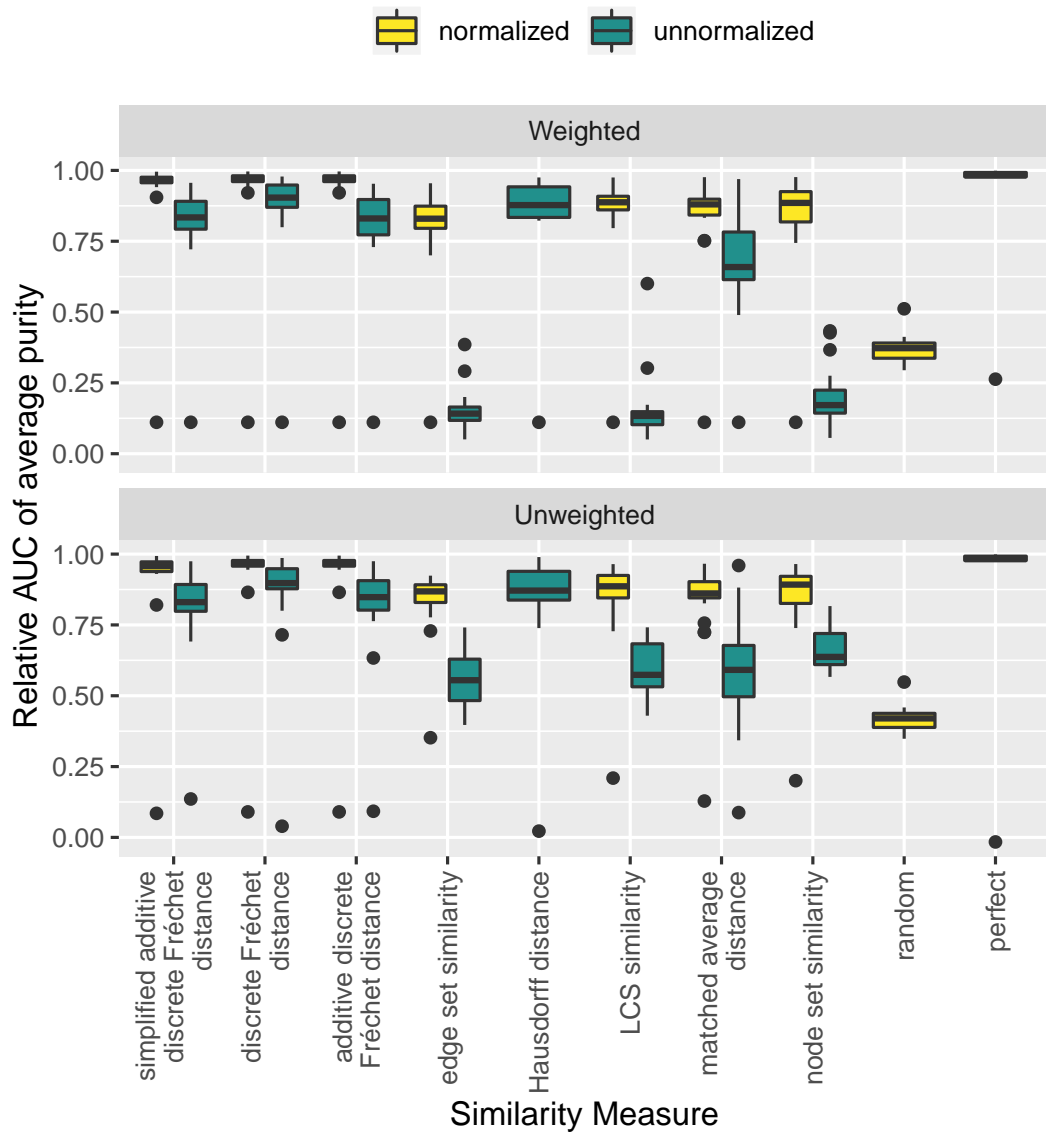


Figure 5.12 Relative AUC of the weighted average purity (top panel) and the unweighted average purity (bottom panel) for all paths of all configurations.

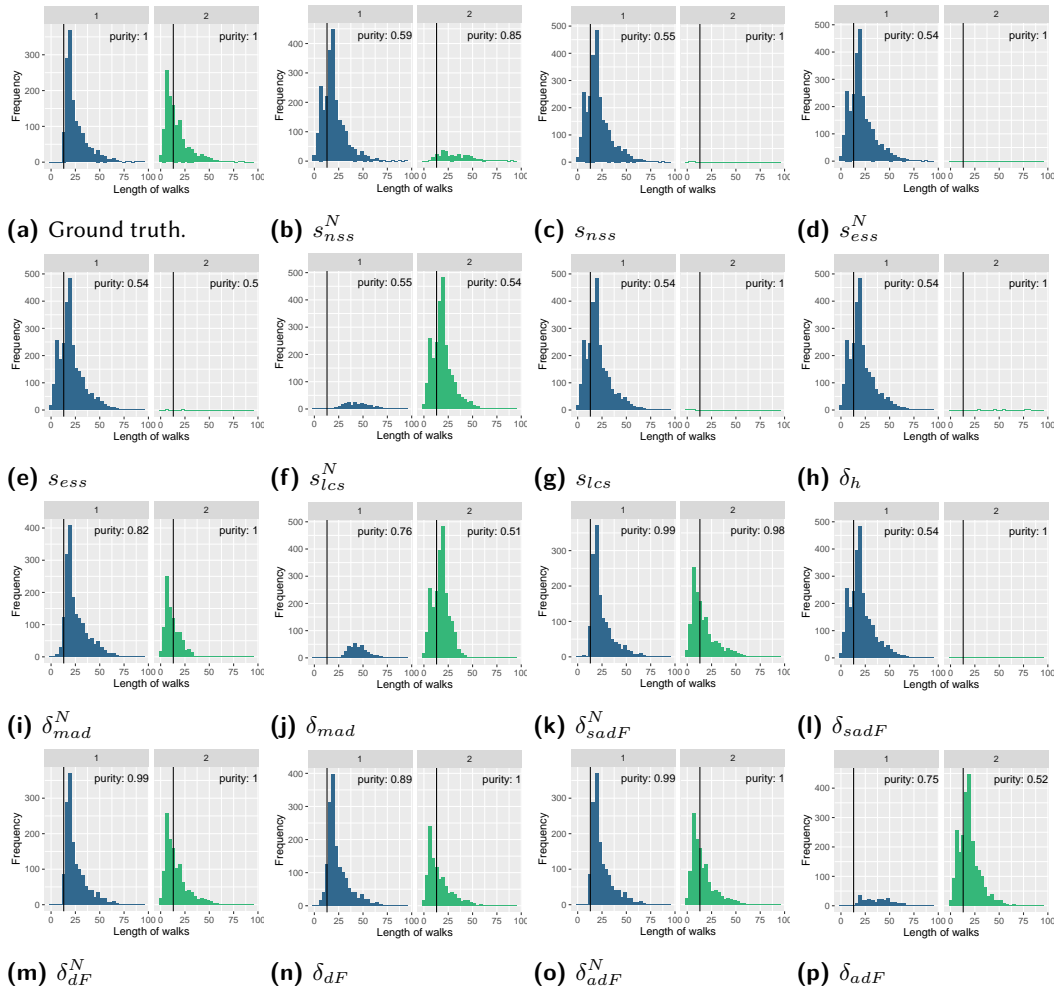


Figure 5.13 For each similarity and distance measure, the clustering with two clusters was selected, and the distribution of walk length within each cluster (clusters 1 and 2) is shown, here for game 723. The vertical line shows the length of the optimal solution for the game. The purity values of the single clusters are shown in the plots. The fraction of solving walks is 0.54 for this game.

The following observations can be made for this game configuration:

- Supporting the previous results, only the clusterings based on the measures δ_{mad}^N , δ_{sadF}^N , δ_{dF}^N , δ_{adF}^N , and δ_{adF}^N (and ground truth based clustering) yielded clusters with approximately the same size. The clusterings based on the remaining similarity and distance measures yielded one clustering with one very small cluster and another containing almost all walks of the dataset.
- Although it is true that a solving walk needs to have a minimum length, it is neither true that non-solving walks are generally shorter than solving walks, nor that the clustering on any of the measures yields a separation of the walks into longer and shorter walks. There are, however, a few game configurations and measures for which the corresponding clustering with two clusters rather separates longer and shorter walks. Figure 5.14 shows examples of these cases. However, it can be seen that the separation by length does not necessarily yield clusters with higher purity.

We therefore repeated the above-described clustering procedure with the same dataset, but only including sufficiently long walks, i.e., walks at least as long as the optimal solution. Thus, for each

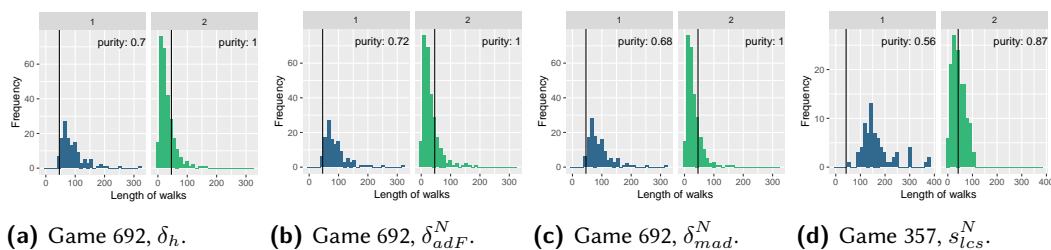


Figure 5.14 Examples of games and clusterings in which (approximately) shorter and longer walks are separated into the two clusters.

configuration c and the corresponding set \mathcal{P}_c , the matrices containing the distance and similarity measures, the ground-truth-based values, and the random numbers, were reduced by removing all rows and columns corresponding to walks shorter than the optimal solution. These reduced matrices were then used as input for the hierarchical clustering procedure as before, and the quality of the resulting clusterings was evaluated by computing their average purity and the Relative AUC of the average purity. Figure 5.15 shows the resulting Relative AUCs aggregated over all configurations. It can be observed that those measures that yielded a high Relative AUC when considering all walks, also yielded a consistently high Relative AUC when only sufficiently long walks were considered: The three normalized Fréchet-based distance measures still yielded a Relative AUC close to 1 on average. Their unnormalized variants performed slightly worse. On the other hand, the unnormalized variants of node set similarity, edge set similarity, and LCS similarity, still yielded a Relative AUC worse than random clustering, but slightly better than for the clustering containing all walks. In general, however, the results for the clustering with all walks and for the clustering with only sufficiently long walks show similar characteristics: Clusterings based on the measures δ_{dF}^N , δ_{adF}^N , and δ_{sadF}^N are best in distinguishing solving and non-solving walks; in other words, exactly those measures that incorporate all three kinds of information contained in a walk.

5.4 Limitations and summary

5.4.1 Limitations

This study has several limitations since it rather serves as a proof-of-concept. We see the following caveats in our work:

Generalizability This study was only performed on one dataset. It is unclear to which extent the results can be transferred to walks from other domains. It is, however, a challenge to acquire suitable datasets with a ground truth regarding group affiliation.

Ground truth used The ground truth we used in the performed experiments contains a large assumption: Similar walks are either both solving or both non-solving, and two walks of which one is solving and the other is non-solving, cannot be similar. This is certainly not true in all cases. However, we are convinced that it is a sufficient approximation for the presented evaluation of the introduced similarity and distance measures.

Finding structurally similar walks Clustering walks as a whole might miss important features of the walks: Two walks might contain very similar (or even identical) sub-paths while being different in other parts. Transferred to the game dataset we used, it is plausible that two players followed the same solution strategy in one phase of the game, but a different strategy in another phase of the game. This cannot be accounted for measures that take two complete walks as input. An improvement of the method could work with *similar sub-paths*: first splitting the walks into segments, then applying similarity measures to the segments,

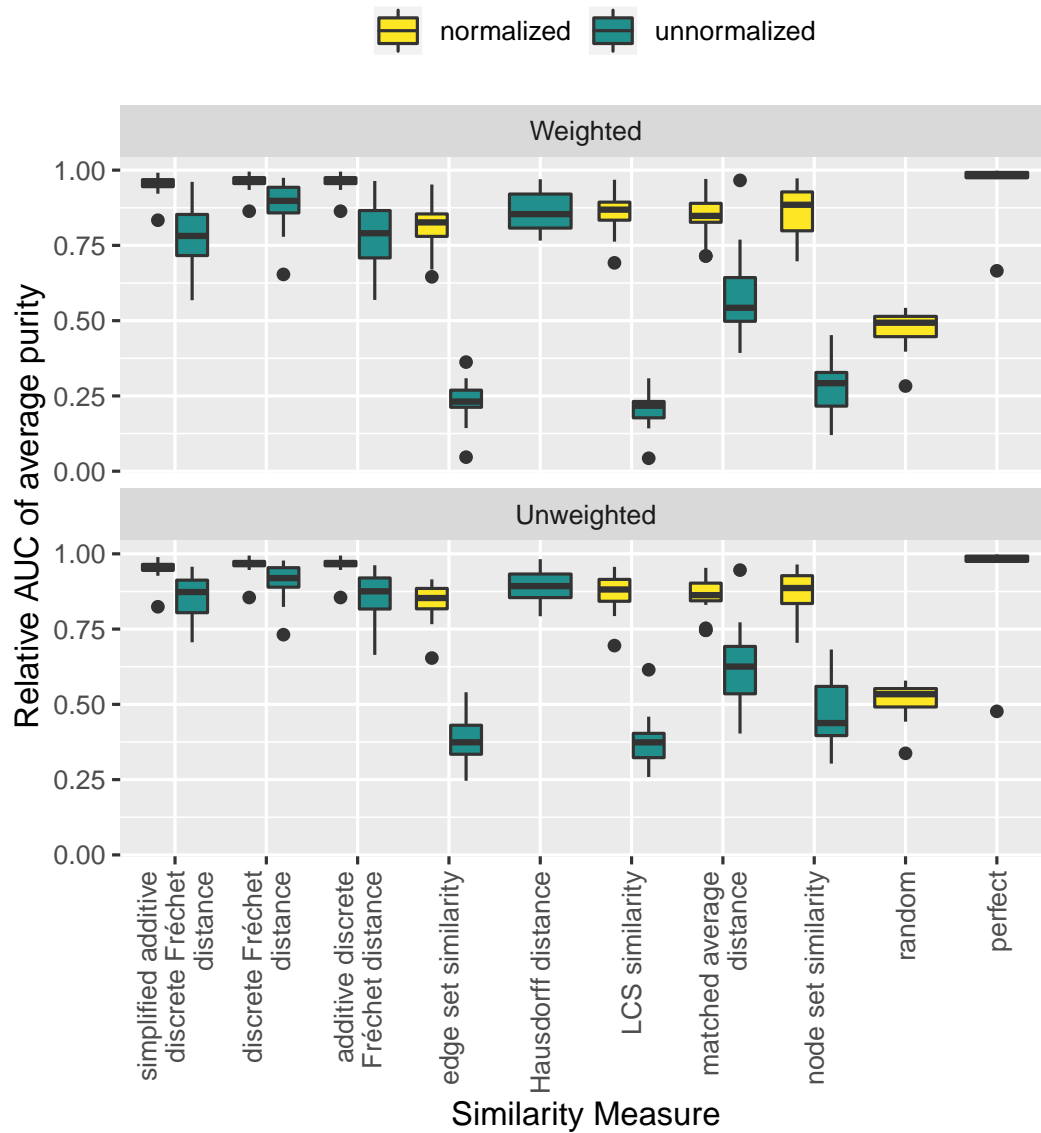


Figure 5.15 Relative AUC of the weighted average purity (top panel) and the unweighted average purity (bottom panel), if only sufficiently long walks are considered.

and finally clustering the walks into groups according to their common or similar sub-paths. It is, however, not clear what would be a meaningful segmentation strategy. An approach for trajectories aimed at finding and clustering similar sub-trajectories is given by Lee et al. [LHW07].

5.4.2 Summary

In this chapter, several distance and similarity measures were introduced with the aim of using them for finding meaningful groups of trajectories. We categorized the introduced measures by the way the walks are modeled: as sets, as sequences, as sets of points in a metric space, or as polygonal curves in a metric space. For each modeling approach, we adapted existing similarity or distance measures for the application on walks. When a walk is understood as a set, set-based measures such as the Jaccard distance for the node or edge set of the walk can be used. When a walk is understood as a sequence, sequence-based distance measures such as the LCS similarity for the sequence of the walk nodes can be used. When a walk is understood as a set of points in a metric space, matched average distance or other measures that compare the positions of the nodes of one walk with the position of the nodes of another walk can be used. When all three kinds of information contained in a walk—its elements, the order of the elements, and the position of the elements in the graph—are used, a walk can be modeled as a polygonal curve in a metric space since the set of graph nodes V together with the graph distance forms a metric space (V, d) (if the graph is connected and undirected). For this type of structure, the discrete Fréchet distance [EM94] can be adapted. We considered an additive version of the discrete Fréchet distance which does not consider the *maximal* distance between the nodes, but the *sum* of the distances.

In order to evaluate the introduced similarity and distance measures for finding meaningful groups of trajectories, we clustered a dataset containing human solutions for a board game according to a standard hierarchical clustering algorithm, using a distance or similarity matrix as input. This procedure was performed for each introduced similarity and distance matrix as well as for two baseline measures, one “perfect” measure serving as upper bound for the clustering quality, and one random measure serving as lower bound for the clustering quality. For the evaluation of the quality of the resulting clusterings, we made the assumption that a clustering based on a good similarity or distance measure should at least be able to distinguish between solving and non-solving walks. We computed the clusters’ purity with respect to the walks’ property of being solving or non-solving, and consider the average purity of the clustering during the complete hierarchical clustering procedure.

We found that the measures based on the Fréchet distance, i.e. discrete Fréchet distance, additive discrete Fréchet distance, or simplified discrete Fréchet distance, yielded clusterings with high average purity with very few clusters. This happened not only for the walks of single games. When we aggregated the average purity of the clusterings of all games and the same distance or similarity measure, we found that clusterings based on Fréchet-like measures yielded consistently high average purity very close to the average purity of the “perfect” clustering.

On the other hand, there are three similarity measures that yielded clusterings with an average purity smaller than that of random clusterings. These three measures—the unnormalized node set, edge set and LCS similarity—are not well-suited for making a distinction between solving and non-solving walks of our dataset.

In order to make sure that the results are not solely based on properties of the walk other than the similarity and distance measures, we repeated the procedure with the same dataset restricted to walks at least the same length as the optimal solution. The reasoning behind this is the fact that solving walks need to have a minimum length while non-solving walks can be arbitrarily short. However, for this subset of walks, the quality of the clusterings was also found to be similar to the

clustering with all walks.

The results imply that similarity or distance measures for walks incorporating all three kinds of information contained in walks—i.e., its contained elements, the order of the elements, and the position of the elements in the graph—are suited best to finding groups of similar walks. Especially those measures that only consider the elements, or only their order, yielded unsatisfactory results for the dataset we examined.

Case Study: Analyzing game trajectories

Chapter outline

In this chapter, the results of an interdisciplinary project with researchers from the field of psychology will be presented. In order to answer the research question from psychology, an experiment was conducted in which participants attempted to solve a Rush Hour game twice—one group with an intervention in between, the control group without an intervention in between. Therefore, for each participant, two solution attempts of the same game instance are available. Common metrics for comparing two solution attempts and assessing their quality are solution length, solution time, and whether the solution attempt was successful. While these are valid metrics for a basic evaluation of the experiment data, they fall short when one wants to understand where and why a solution attempt failed: For this, more detailed analysis methods are needed. At this point, it is beneficial to consider a participant's solution as a trajectory in the state space of the game and consider the performed game moves in detail, particularly those moves that were not optimal. For this reason, in this chapter, we propose an error category system that is able to classify these error moves according to the (assumed) motive of the participant. Given a participant's solution attempt, the error category system automatically labels each error move according to the assumed reason why the error move occurred. This approach allows answering questions such as: Which types of errors are persistent through several solution attempts—and which can be avoided by the participant in a second attempt? When considering in which situations an error type is actually *possible*, it can be checked whether there are error types that occur more often than expected. Besides the error category system, we also propose a method for identifying similar error moves in different solution attempts. This is not trivial since two error moves do not need to occur in identical game situations in order to be considered as conceptually equal. Using the proposed matching procedure for error moves, it is possible to analyze which factors are essential to enable a participant to avoid an error in the second solution attempt—and which factors cause a repetition of the same error. While the presented error category system and the matching procedure for error moves is only used to answer these questions specifically for the game Rush Hour, we believe that almost all error categories and the matching procedure can be transferred to other types of tasks such that the insights and methodological approaches presented in this chapter are also valuable for other learning contexts.

This chapter is based on [1] and presents work done jointly with Olaf Peters and Susanne Narciss.

6.1 Motivation and background

The background of the described collaboration project is learning psychology, i.e., the issue of what can help a student to learn (more) effectively. Specifically, the researchers around Susanne Narciss at the Technical University of Dresden are interested in the effects of formative feedback in the learning process. Formative feedback is “information communicated to the learner that is intended to modify his or her thinking or behavior for the purpose of improving learning” [Shu08]. Sources of formative feedback might be a teacher, a system, another learner, or the learner him- or herself (internal feedback). Usually, the research focus is on the *receiver* of the feedback [CM10, SND10]. In this project, however, the focus was on the *feedback provider*. By revising a solution of another person (a “peer”) and generating feedback, the learner is required to reflect on the solution, detect errors and provide suggestions for improving the solution. This might have a positive effect on the learner’s performance when solving the task him- or herself. The research question of the collaboration project was therefore: Does generating peer feedback have an effect on one’s own performance in solving a task? A second minor research question was whether the peer’s confidence regarding the correctness of their solution has an impact on the solving performance.

In order to investigate this question, an experiment was designed in which participants were randomly assigned to two groups (for a detailed description of the experimental procedure, see Section 6.3). All participants first attempted to solve a task. The participants of the first group were then asked to generate feedback on another (fictitious) participant’s solution of the same task, whereas the participants of the second group did not perform this feedback generating process. All participants then attempted to solve the same task again.

The reason why this collaboration project is of relevance for this thesis is the lack of detailed methods for analyzing, evaluating the quality, and comparing several solution attempts of a task. For each participant, two solution attempts of the same task are available and for answering the research question, questions such as the following are of interest: Did the participant improve in the second solution attempt? Which errors persisted in the second solution attempt? Are there significant differences in the performance of the two experimental groups? Can situations be identified in which participants who generated feedback before the second solution attempt performed better than participants who did not see a peer’s solution? To avoid an error in the second attempt, is it necessary to be aware of the error?

The challenge is thus to develop appropriate methods for analyzing such data. An important prerequisite for a detailed analysis of the data is an appropriate choice of the task. In problem-solving research, a distinction is often made between well-defined and ill-defined problems [NS72, And93, Gre78]. For well-defined problems, the initial situation, the goal situation and all applicable operators for transforming the situation are known. For ill-defined problems, at least one of these is not available to the problem solver. For a quantitative evaluation of the quality of a solution, only well-defined problems are possible as a task for the experiment.

We selected the board game Rush Hour (described in Chapter 5) as a task, mainly for two reasons:

- It is well-defined and sequential, i.e., it cannot be solved within a single step, but requires a sequence of actions to solve a task. This has two advantages: First, since the goal situation is clearly defined, each move is either correct or not. This is beneficial for assessing the quality of a solution. Second, since solving the task requires more than one step, there is a wide range of different solutions—and not only one correct and one wrong one. If there is only one correct and one wrong solution, generating peer feedback and investigating its effect is difficult to impossible.
- Although the task is easy to understand and all elements, rules, and principles are presented to the learner, there are game configurations of Rush Hour ranging from medium to high difficulty [RSF11, BZ15]. In order to see an improvement in solving performance, it is necessary

to choose a task that most students will not solve optimally in the first attempt.

Like any well-defined sequential task, a Rush Hour game configuration induces a state space where nodes represent board configurations and edges represent legal game moves. A solution is thus a trajectory from the node representing the start configuration to a node representing a goal state. For Rush Hour, all moves are reversible which means that the state space can be modeled as an undirected graph, and, if the initial configuration is solvable, a goal state can be reached from any node of the state space. This can be different for other types of games where the player can end up in a “dead end” with a wrong move. Because this cannot happen for Rush Hour, it holds that in every node of a state space, there exists at least one *correct* move that decreases the distance to a goal state. Furthermore, due to the reversibility of the moves, for each move, it holds that it can increase or decrease the distance to the closest goal state by at most one step. We can therefore group the moves into three types of moves: *correct* moves which decrease the distance to a goal state by 1, *wrong* moves which increase the distance to a goal state by 1, and *unnecessary* moves which neither increase nor decrease the distance to a goal state. We will refer to both, wrong and unnecessary moves, as error moves because a solution containing at least one of these moves cannot have the optimal length anymore. A wrong move is worse than an unnecessary move because it increases the length of the solution by two as it needs to be corrected by an additional move, while an unnecessary move only lengthens the solution by one.

When considering several error moves, it becomes obvious that they can occur for different reasons: A participant might, e.g. have a wrong understanding of the task or might make a mistake in planning the solution strategy; however, an accidental wrong click can also lead to an error move. Our hypothesis is that there exist conceptually different error moves. For the analysis of human solution attempts of a Rush Hour game or any other sequential well-defined problem, it is beneficial to detect the different types of conceptual errors: In most experimental research projects concerned with human problem-solving, human solution attempts of a problem need to be analyzed. To assess the quality of a solution attempt, simple metrics are applied in most cases, such as solving success (was the problem instance successfully solved?), solving time (how long did it take to solve the problem instance?), or solution length (how many steps were needed to solve the problem instance?). In order to understand the cognitive processes that lead to a specific solution of a problem, it is necessary to analyze the human solution in more detail. Focusing on the error moves has great potential since those are the points where “something goes wrong”: Identifying situations that are prone to errors can for example help to understand the difficulty of problems. For this purpose, however, an error move should be rated differently in terms of whether it is a conceptual error or an unintentional error. Therefore, a categorization of error moves is needed that enables classifying error moves with respect to their (assumed) motive. At a task-independent level, Norman and Reason categorize error moves into *planning failures* (mistakes) and *execution failure* (slips) [Rea90]: “An error in the intention is called a mistake. An error in carrying out the intention is called a slip.” [Nor83] However, in order to understand the process of human problem-solving, a more detailed categorization is needed. More detailed categorizations do exist for specific tasks: Caramazza et al. [CMVR87] considered types of errors in word-spelling tasks, Rasmussen categorized human errors in the early days of human-computer interaction [Ras82]; several approaches consider human errors in patient medication in hospitals [Lea94, ZPJS04], conceptual errors in learning mathematical concepts [ENSM12], or in performing a proof [ADS12]. There exist, however, very few approaches for categorizing errors in well-defined sequential problems, such as the categorization of error moves in a block design puzzle [TS04]. For Rush Hour, to the best of our knowledge, the present work is the first extensive categorization of error moves. The following paragraphs will introduce a category system for error moves. While their concrete definition is phrased for the Rush Hour game, most of them are based on general principles that can also be applied to other types of well-defined sequential tasks.

The remainder of this chapter is structured as follows: In Section 6.2, we will present the error category system, then, as a validation of the proposed system, report on its application to a large

dataset containing more than 31000 human solutions of Rush Hour tasks with a total of more than 190000 error moves to be categorized. Section 6.3 describes the experiment conducted to answer the research question from the field of psychology regarding whether generating peer feedback has an effect on a learner's performance. Section 6.4 presents the application of the error category system to the experimental data. Here, the value of such a system is shown since it enables a more detailed analysis of the experimental data. Section 6.5 provided a short summary of the presented work.

6.2 Error categorization system

We will introduce eleven error categories, with four being rather descriptive and seven being based on the actual board configuration which we call conceptual error categories.

6.2.1 Descriptive error categories

The following four error categories are called descriptive because they describe the pattern of the error move. Their definition is independent of the board configuration and to a certain extent even independent of the actual game since only their pattern in the state space is needed for their detection.

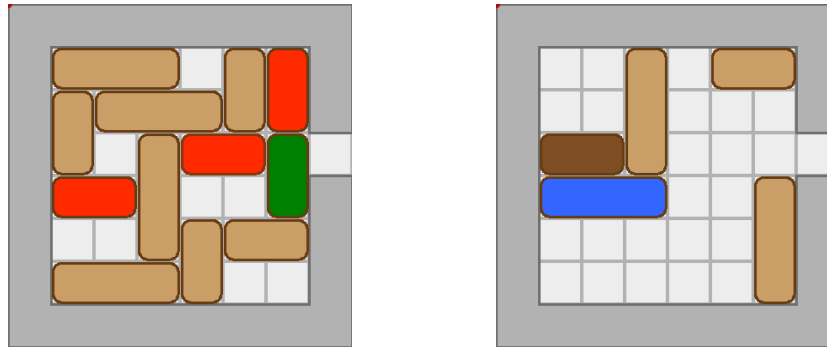
Generalized Undo Mistake It is never optimal to reverse the effect of a move: A solution containing the moves (v, w) and (w, v) cannot be optimal anymore and at least one of these two moves is an error move. We thus call an error move (v, w) a *Generalized Undo Mistake* if the solution contains both (v, w) and (w, v) . If both moves are error moves because they are both unnecessary moves, they are both categorized as such a mistake. Note that the two moves do not need to occur consecutively in the solution.

Undo Mistake If a solution contains (v, w) and (w, v) consecutively, the error move is additionally categorized as *Undo Mistake*.

Two In One Mistake If a solution contains the move sequence $(v, w), (w, x)$ and the state space contains the edge (v, x) , at least one of the two moves is an error move because the effect of the two moves could have been accomplished by only one move. Thus, we call such an error move a *Two in One Mistake*. In the case of Rush Hour, the two moves need to involve the same car, otherwise the edge (v, x) would not exist. The existence of such an error move shows that the participant thought that the first move needed to be corrected or adjusted by a second move.

Equivalence Trap A solution might contain moves that “do not make a difference” because they do not change the relevant relations of the board configurations. For Rush Hour, the relevant relations of the board are the blockings of the cars: Which car does, in its current position, restrain the movement of which other car? If a move is performed that does not change these blocking relations, the move does not have any effect on the game situation: In both situations, before and after the move, exactly the same set of moves is possible. We formalize this observation by an equivalence of board configurations. We say that *a car A blocks a car B* in a board configuration if

- (i) A and B are both placed horizontally (or both vertically) on the board and are in the same row (or column) of the board, or
- (ii) A is placed horizontally and B is placed vertically on the board, and A occupies cells in the column in which B is placed, or
- (iii) A is placed vertically and B is placed horizontally on the board, and A occupies cells in



(a) Concept of blocking cars: In the depicted game configuration, the red cars are blocked by the green car.

(b) Equivalence Trap: Moving the blue car will lead to an *Equivalence Trap* mistake.

Figure 6.1 Example board configurations illustrating the concept of equivalent game configurations.

the row in which B is placed.

Simply put, if A was not there, B could move further on the board (if all other cars are ignored). Figure 6.1a shows an example: The green car blocks the three red cars. We say that two game configurations are equivalent if each car blocks exactly the same set of cars in both configurations. With this concept, we call an error move (v, w) an *Equivalence Trap Mistake* if v and w are equivalent. Thus, a move of this type does not have any relevant effect on the game configuration because all cars are still blocked by the same cars as before. In Figure 6.1b, moving the blue car yields an error move categorized as *Equivalence Trap* mistake. While this definition of equivalent game configurations is specific for Rush Hour, the general concept of equivalent situations can also be applied to other games or well-defined sequential problems, meaning that all descriptive error categories can be easily applied to any well-defined sequential problem.

6.2.2 Conceptual error categories

While the presence of an error move categorized as one of the descriptive error categories shows that the participant's solution is somehow erroneous, this does not allow any conclusion as to why an error occurred. The following seven error categories are based on the actual game configuration and presume a motive for the mistake. Three categories are based on spatial features of the board, four are related to the goal of the game. The underlying general principles of the error categories are generalizable to other games.

Categories based on spatial features

Most tasks have some visual representation. The following three error categories are based on spatial features of the visual representation.

Stay Local Mistake When confronted with a complex task containing several interdependencies, a good heuristic for reducing the complexity of the task is to split it into subproblems and resolve the subproblems one after the other. For spatial representation, this means that structures on the board are considered locally and resolved one by one. This is often a successful strategy, but sometimes it fails. We hypothesize that this heuristic leads to error moves where the participant considers the same local structure as in the previous move, but a change to a different local structure would have been necessary for an optimal solution. For Rush Hour,

Table 6.1 Examples of the introduced conceptual error categories: Correct moves are shown with a green frame, error moves with a red frame.

Error category	Example move	Correct alternatives
Stay Local		
Relaxed Car Unit		
Border Attraction		
Early Unblock		
Avoid Blocking The Exit		
Early Target Car Move		
Avoid Moving Target Car Backwards		

this yields a simple solution heuristic: Check which board cells have been freed by the previous move, then move another car onto the freed cells. If this heuristic fails, i.e. leads to an error move, this error move is categorized as *Stay Local Mistake*. Formally, an error move $m = (v, w)$ moving car i is called a *Stay Local Mistake*, if

- (i) in configuration w , car i occupies at least on cell that was freed by the move directly before m , and
- (ii) in configuration v , there exists no correct alternative move that also uses one of these freed cells.

Table 6.1 shows an example of such an error move: The first (correct) move frees a cell in the bottom row. The depicted error move uses the freed cell by moving the horizontal car in the bottom row, whereas for a correct alternative move, a different car would have needed to be moved.

Relaxed Car Unit Mistake Bennati et al. [BBRK14] found in experimental data that the existence of so-called clusters of cars on a Rush Hour board has an effect on the optimality of the participants' solutions. A cluster of cars (or car unit) consists of two cars, either both horizontally or both vertically, which are placed in directly consecutive rows or columns, and at least one cell occupied by the first car is adjacent to at least one cell occupied by the other car. In Figure 6.1b, the blue and the dark brown car form a car unit. Bennati et al. show that the presence of such a cluster decreases the chances of a participant to find an optimal solution. They explain this finding with Gestalt effects: Humans tend to perceive close objects as belonging together which is why they are treated similarly. Following this argument, for the case of Rush Hour, two close cars are perceived as a unit and are moved together—which might not always be a correct move. Based on this observation, we define a new error category called *Relaxed Car Unit Mistake*¹: It means that a participant moves both cars of a car unit in consecutive moves although only one needed to be moved in this situation. Formally, an error move $m = (v, w)$ is called a *Relaxed Car Unit Mistake* if there is another move m' directly before or after m in the solution, thus $m' = (x, v)$ or $m' = (w, y)$, such that

- (i) in the configuration before m and m' and in the configuration after m and m' , there exists a car unit, and
- (ii) m and m' involve moving the cars of the car unit in the same direction, and
- (iii) in the configuration before m and m' , there does not exist any alternative sequence of two correct moves involving the same two cars.

Thus, moving one car of the car unit (possibly in a correct move) provokes an error move in order to retain the car unit. Note that the order of the two moves is not relevant for the categorization of this error type: Whether a correct move is followed by an error move or vice versa, yields the same categorization. This is in accordance with the underlying idea of this error type: If the cars are perceived as a unit, it is irrelevant for the participant which of them is moved first. Table 6.1 shows an example: The two horizontal cars in the top two rows form a car unit and are moved together. While the second move is correct, the first move is a wrong move. It would have been correct to only move the lower car of the car unit instead. While the concept of car units is specifically defined for Rush Hour, the principle of objects perceived as a unit can also be transferred to other games or tasks for which a similar error category can then be derived.

Border Attraction Mistake When playing Rush Hour, it feels natural to move a car as far as possible because such a move is seemingly better than only moving the car by only one or two cells: It creates the impression of creating more space on the board which appears to be

¹The prefix *Relaxed* indicates that a perfect car unit in which the two cars are perfectly aligned is not necessary. Rather, a car unit where one cell of overlap is sufficient is used.

better for the further course of the game. Error moves resulting from applying this heuristic are called *Border Attraction Mistakes*: An error move $m = (v, w)$ is categorized as *Border Attraction Mistake* if

- (i) the moved car is moved as far as possible (constrained by the border of the board or by another car), and
- (ii) there exists a correct alternative move that moves the same car in the same direction, but less far than m .

In other words, it was the correct choice to move a particular car into this direction, but moving it that far yields an error move. Table 6.1 shows an example.

Goal-based categories

Any well-defined problem-solving task has a goal to reach. Thus, a simple heuristic for solving a task is to transform the current task configuration into another configuration that appears more similar to the goal configuration, whenever possible. This heuristic is so simple that it will fail in most cases (and is not suited as a stand-alone heuristic since it is not clear which step to take if the heuristic is not applicable). We still hypothesize that humans will prefer those moves that (seemingly or actually) yield a configuration more similar to the goal configuration, and avoid moves that (seemingly or actually) yield a configuration less similar to the goal configuration. Errors resulting from those types of moves will be described in the following and can be twofold: when making a move that seems to yield a configuration more similar to the goal configuration is not a correct choice at that point (but might be later in the course of the game), or when avoiding a move that seems to yield a configuration less similar to the goal situation is necessary at that point in order to get closer to the goal state.

For Rush Hour, the goal can be formulated in two ways: The target car needs to be moved towards the exit, or all cells to the right of the target car need to be freed. For each version of phrasing the goal, we get an “Avoid”-mistake and a “Too Early”-mistake.

Avoid Blocking the Exit Mistake Moving a car onto a cell to the right of the target car and thus additionally blocking the target car from the exit seems counterintuitive for reaching the goal of the game and might be avoided. An error move resulting from the avoidance of such move where it would be necessary is called an *Avoid Blocking the Exit Mistake*. Formally, an error move is categorized as *Avoid Blocking the Exit Mistake* if

- (i) neither before nor after the move, the moved car occupies any cell to the right of the target car in its row, and
- (ii) all correct alternative moves require moving a car onto a cell to the right of the target car (and thus blocking the target car).

Hence, it would have been necessary to block the target car, but this move is avoided and an error move is made instead. Table 6.1 shows an example.

Early Unblock Mistake The corresponding “Too Early”-mistake involves moving the target car towards the exit although it is not necessary—even counterproductive—at this point in the course of the game. Thus, an error move is categorized as *Early Unblock Mistake* if

- (i) the moved car blocks the exit before the move, and
- (ii) does not block the exit after the move, and
- (iii) no correct alternative move exists in which the same car is moved in the same direction as in the error move.

The last requirement ensures that the move shown in Table 6.1 as an example for a *Border Attraction Mistake* is not categorized as an *Early Unblock Mistake* because there exists a correct alternative that also unblocks the target car.

Avoid Moving Target Car Backwards Mistake Similarly to avoiding moves that block the exit, we define an error move as *Avoid Moving Target Car Backwards Mistake* if the participant seems to avoid a move that moves the target to the left, i.e., further away from the exit. An error move $m = (v, w)$ is categorized as *Avoid Moving Target Car Backwards Mistake* if

- (i) it does not move the target car to the left, while
- (ii) all correct alternative moves do.

Thus, a correct move in v requires the target car to be moved to the left, but instead an error move is made.

Early Target Car Move Mistake In the same scheme as the *Early Unblock Mistake*, an *Early Target Car Move Mistake* is present if the target car is moved to the right (towards the exit) although this is not correct at this point in the course of the game. Thus, an error move is categorized as *Early Target Car Move Mistake* if it moves the target car to the right (while no correct alternative move does so).

The error category system is implemented such that, given the initial game configuration and a participant's solution, it is possible to automatically label each error move with the corresponding categories.

6.2.3 Validation of error category system

In order to validate the proposed error category system, we applied it to the dataset described in Chapter 3, containing 56 Rush Hour game configurations and students' solution attempts for these game configurations. The data was collected by an online learning platform developed by a Czech university [JP12]. For the validation of the error categories, we excluded unsolved attempts and attempts that needed three times more steps than the optimal solution. The considered data then contained a total of more than 31 000 solution attempts for 56 game configurations, with between 13 and 2319 solutions for a single game. The difficulty of the game configurations varied, with an optimal solution length range between 3 and 50 moves. In these 31 000 solutions, more than 190 000 error moves were observed. The number of observed error moves varied across the different game configurations. Figure 6.2 shows for each game configuration the number of observed error moves per solution attempt. The order of the game configurations on the x -axis is due to the length of their optimal solution length; thus, easier games (with a shorter optimal solution) are on the left and harder games are on the right. Not surprisingly, the solutions of the harder games contained a considerably larger number of error moves, although the variance is very large. The average number of error moves per solution ranges from 2.1 ± 1.2 error moves for a game configuration with an optimal solution length of 3 to an average number of 41.3 ± 12.5 for a game configuration with an optimal solution length of 47.

Although the data was collected in an uncontrolled environment (since the games are browser-based and publicly available), the dataset is still suited as a benchmark dataset for testing whether the proposed error category system is a reasonable categorization of common errors in playing Rush Hour.

All error moves were categorized according to the introduced category system. To validate the category system, we checked:

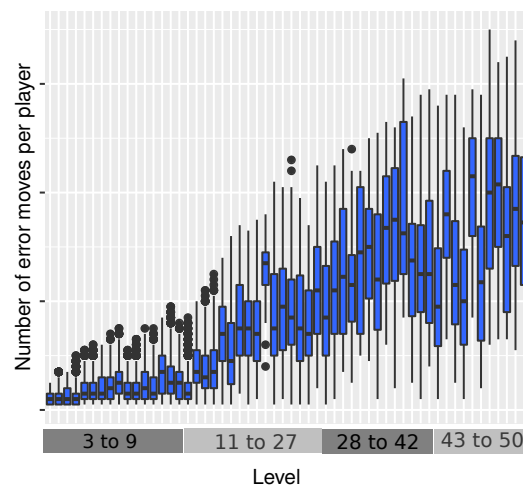
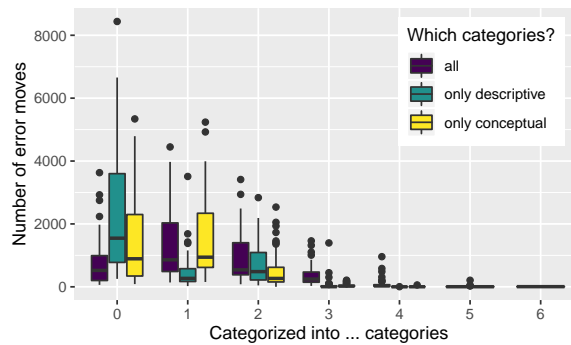


Figure 6.2 Validation of error category system on benchmark dataset: The figure shows the number of observed error moves for each game configuration. The game configurations on the x -axis are ordered by the length of their optimal solution: Easier games are on the left, harder games on the right. The numbers in the gray boxes show the length of the optimal solution.

- (i) Do the categories cover the majority of observed errors?
- (ii) Do the categories overlap?
- (iii) Do any unnecessary categories exist?

Do the categories cover the majority of observed errors? Figure 6.3 shows for the benchmark dataset which percentage of observed error moves falls into which number of categories. We found that for 23 % of all error moves, no error category applied, while the remaining 77 % fell into at least one category. If only the conceptual categories were considered, 40 % of the error moves remained uncategorized. However, it seems that especially for the easier games, the category system needs to be refined. If only games with an optimal solution of at least 13 steps were considered, the percentage of uncategorized moves dropped to 17 % (with all categories) and 33 % (only conceptual categories)—although far more error moves were observed in these games. Figure 6.4 supports this observation: For each game, its optimal solution length and its percentage of uncategorized error moves is shown, separately for all categories, for both the conceptual and the descriptive categories. It can be seen that the percentage of errors not categorized by any conceptual category is rather constant for harder games, but for easier games, there is a large variation and the percentage is higher on average. When the proportion of uncategorized error moves was considered separately for each of the 56 games (not shown), the proportion ranged from 7 to 52 %; for games with an optimal solution length of at least 13, the proportion ranged from 7 to 27 %. There was, for example, one game with more than 1 000 solutions and more than 8 000 observed error moves, of which only 1 000 moves were uncategorized. Of the categorized error moves of this game, only 700 are solely characterized by descriptive categories; the rest (more than 7 000) fall into at least one of the conceptual categories.

Do the categories overlap? Since 83 % of all error moves of the less easy games of the benchmark dataset fall into at least one category, the question arises whether there are many error moves that fall into more than one category. This is more acceptable for the descriptive categories than for the conceptual ones. Figure 6.3 shows that only 15 % of all error moves fall into more than one conceptual error category, and only 1 % fall into more than two conceptual categories. We found that multiple categorizations were mainly due to *Stay Local Mistakes*: If this error category is excluded, the percentage of errors categorized into multiple categories drops to only 6 % which is a reasonably small value.



(a) Validation of error category system: The plot shows the absolute frequency of error moves categorized into number of categories; separately for all categories, only for the descriptive categories and only for the conceptual categories.

Games	in ... categories				
	0	1	2	3	> 3
all	23 %	37 %	27 %	10 %	< 3 %
less easy	17 %	38 %	29 %	12 %	< 4 %

Games	in ... conceptual categories			
	0	1	2	> 2
all	40 %	44 %	15 %	1 %
less easy	33 %	48 %	17 %	2 %

(b) Percentage of the 194 290 observed error moves in the solutions falling into exactly 0, 1,... categories, either for all game configurations or for those with an optimal solution length of at least 13 (for which 89 540 error moves were observed).

Figure 6.3 Validation of the introduced error category system, showing how many of the observed error moves fall into how many categories.

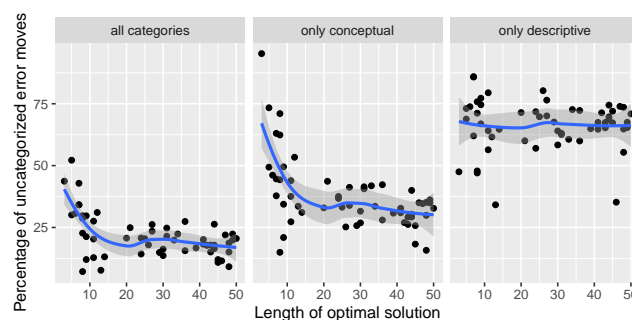


Figure 6.4 Validation of proposed error category system: The figure shows the percentage of uncategorized error moves of the benchmark dataset, dependent on the difficulty of the game, i.e. its optimal solution length. Each point represents one game configuration.

Table 6.2 Application of the error category system to the benchmark dataset. The tables help to show whether there are unnecessary categories in the category system.

Games	Category	is subset of
All	Undo	Gen. Undo
Game 266	Stay Local	Early Unblock
Game 342	Undo	Early Unblock
Game 342	Gen. Undo	Early Unblock
Game 566	Stay Local	Border Attraction
Game 674	Stay Local	Border Attraction

(a) The table shows in which game configurations of the benchmark dataset, a category is a subset of another one.

Error category	not contained in
Border Attraction	13/56 game configurations
Avoid Blocking The Exit	9/56 game configurations
Avoid Moving Target Car Backwards	8/56 game configurations
Early Unblock	3/56 game configurations
Early Target Car Move	1/56 game configurations

(b) The table shows in how many game configurations of the benchmark dataset, error categories do not occur at all in the solutions.

Are there unnecessary categories? In almost all tasks, error moves of each category occurred. Only for a few game configurations did not all error categories occur. For example, *Border Attraction* Mistakes did not occur in 13 of the 56 game configurations (see Table 6.2b). However, most categories are contained in the solutions of all or almost all games. Furthermore, except for the pair *Undo* and *Generalized Undo*, there is no pair of categories such that an error move of the first category is always one of the other category as well. There are, however, a few game configurations for which this is the case (see Table 6.2a). This involves only the error types *Undo*, *Generalized Undo* and *Stay Local* which are rather general categories by design.

It can be summarized that the introduced error category system seems to be well suited for categorizing existing error moves in Rush Hour solutions—although the category definitions are strict, and the variety of different game situations is large. There is, however, still potential for improvement because about a third of the observed error moves was found not to be covered by any conceptual category.

In the following, we will use the proposed error category system for a detailed analysis of the data from the conducted experiment. Therefore, in the next section, we will describe the experimental setup before presenting the analysis of the experimental data based on the error category system in Section 6.4.

6.3 Experimental setup

In order to investigate the research question posed by the group of psychologists, an experiment with 138 participants was conducted at the Department of Psychology at the Technical University of Dresden.

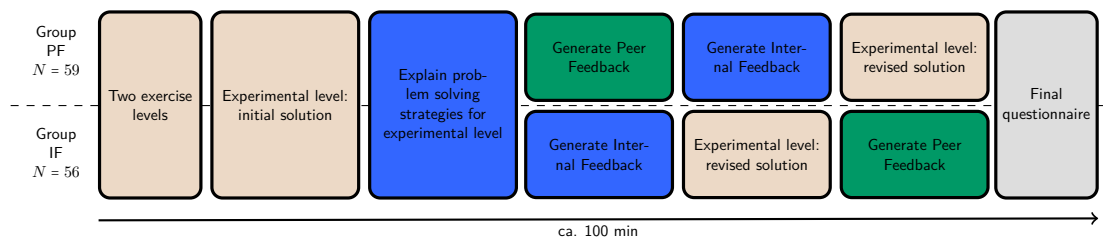


Figure 6.5 Design and procedure of the experiment. The different phases were done in the depicted order from left to right. Each row represents one experimental group.

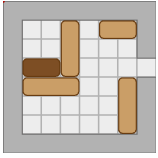
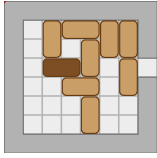
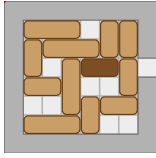
For each participant, the experiment session lasted approximately 100 minutes and consisted of six main phases, with a different order of the phases depending on the group (cf. Figure 6.5): All participants attempted to solve two rather easy exercise game configurations of Rush Hour in order to get acquainted with the rules of the game. After the exercise phase, all participants attempted to solve a Rush Hour task of medium difficulty, shown in Table 6.3 (length of optimal solution: 25 moves) and referred to as the experimental level. The solution for the experimental level is referred to as the initial solution. While there was no limit on time or on the number of moves for the first exercise task, the second exercise task had to be solved within 10 minutes and 50 moves, the experimental level within 10 minutes and 60 moves. The participants were allowed to undo their last move (but only one at a time; reversing the last move through a regular move is clearly always possible). Moves undone by a participant via the Undo button were not included in the participant's solution in order to prevent the inclusion of accidental moves. Rush Hour tasks were solved in a computer-based experiment environment.

After the first solution attempt, all participants were asked to document the problem-solving strategy they employed in their initial solution attempt and had a phase of generating internal feedback on their own solution. Participants of the experimental group additionally had a phase of generating feedback on a (fictitious) peer's solution before they attempted to solve the experimental level for a second time (referred to as *revised solution*). The participants of the control group did not have this phase of peer feedback generation between the initial and the revised solution. The peer's solution solved the task, but was not optimal: It contained four wrong moves and four unnecessary moves which resulted in a total solution length of 37 steps. All error moves fell into at least one error category; five conceptual and two descriptive errors were contained in the peer draft. The participants were asked to generate feedback on the peer's solution guided by the following aspects for each section: (i) correctness, i.e., optimality of the section, (ii) location of errors, (iii) kind of errors and consequences for the remaining solution (while they were not aware of the error categories introduced in Section 6.2), and (iv) suggestions on how to improve the solution approach. For the second solution attempt, the participants were, as in the initial attempt, restricted by 60 moves and 10 minutes time. The participants had all the materials and documents from previous phases at hand while solving the task.

The participants were assigned randomly to one of four conditions in a 2×2 factorial design with the following factors: (i) generating peer and internal feedback prior to revision of the task (PF) versus generating internal feedback prior to revision task (IF), and (ii) high peer response confidence (High RC) versus low peer response confidence (Low RC). Since the focus of this work is on methodological approaches for analyzing and comparing human solution approaches, and the analysis will mainly focus on the comparison of each participant's first and second solution attempts, the following analysis and discussion will either be restricted to two groups (IF and PF) or neglect the groups².

²The results of statistical analyses testing for group differences to answer the research question from psychology will appear in a separate publication.

Table 6.3 Tasks used in the experiment.

	First exercise task	Second exercise task	Experimental level
			
Difficulty rating by manufacturer	Beginner (1 of 5 difficulty categories)	Beginner (1 of 5 difficulty categories)	Intermediate (3 of 5 difficulty categories)
Optimal solution length	9	13	25
Size of state space	$ V = 102, E = 282$	$ V = 7223, E = 41384$	$ V = 3182, E = 13013$

6.4 Analysis of experiment results

Conducting the described experiment yielded two solution attempts for the experimental level for each of the 138 participants³. Due to incomplete data, the results of 23 participants had to be excluded from the analysis. Thus, the experimental data of 115 participants (95 female, 18 male; age: 18 to 30 years, $M_{age} = 20.88 \pm 2.80$) were used in the analysis. We found that in the initial attempt, 70 participants managed to solve the task (and 45 failed to solve it within the given time frame or the maximal number of moves). In the second solution attempt (also referred to as *revised solution* in the following), 108 participants were able to solve the task, while seven also failed in the second attempt. Notably, neither in the first nor in the second attempt was a single participant able to solve the game within the optimal number of moves. For both solution attempts, the shortest solution found by any participant contained 27 moves while the task can be solved within 25 moves.

This shows that neither the percentage of participants able to solve the task at all nor the percentage of participants able to solve the task within the optimal number of moves, is a sufficient metric for a detailed analysis. When considering the error moves made by the participants, we found that the participants' initial solutions contained on average 9.6 ± 8.0 wrong moves and 8.0 ± 4.2 unnecessary moves while these numbers dropped to 2.6 ± 4.1 wrong moves and 3.5 ± 3.6 unnecessary moves in the revised solution. We applied the proposed error category system to all error moves contained in all solution attempts. Table 6.4 shows the fraction of error moves classified into the number of categories, separated by initial and revised solution attempt, once for all error categories and once only for the conceptual categories. For this task, the categorization of error moves worked well: Only 7% of all error moves were left uncategorized while the number of error moves categorized into multiple categories is reasonably small. It can be seen that a considerable portion of the latter can be explained by the fact that they fall into (at least) one conceptual and (at least) one descriptive error category. When we only consider categorization by conceptual categories, the percentage of multiple categorizations drops from 58% to 32%.

We will structure the following paragraphs along the following questions:

³and each participant's solution of two exercise tasks, their generated internal feedback and peer feedback. While including this information in the analysis is also interesting, in this chapter, we focus on the application of the error category system to the experimental data.

Table 6.4 Experimental data: Each of the 2026 error moves in the initial solutions and the 661 error moves in the revised solutions of the experimental level was categorized according to the proposed error category system. The table shows how many of the error moves were categorized in how many categories (conceptual and descriptive).

		categorized into ... categories					
#		0	1	2	3	4	5
Initial	2026	7 %	35 %	35 %	18 %	4 %	<1 %
Revised	661	7 %	54 %	27 %	9 %	3 %	<1 %

		categorized into ... conceptual categories					
#		0	1	2	3	4	5
Initial	2026	15 %	54 %	26 %	5 %	<1 %	0
Revised	661	15 %	63 %	18 %	3 %	<1 %	0

- (i) How often does an error type occur in the solution attempts of the experimental level?
- (ii) In which situations do the error types occur?
- (iii) Are there error types that are often possible, but rarely made or—the other way around—rarely possible, but often made in these situations?
- (iv) Which error types are avoided in the revised solutions? On which factors does it depend whether or not an error can be avoided?

6.4.1 How often does an error type occur in the solution attempts?

We first consider the occurrence frequency of each error type in the participants' initial and revised solutions. Figure 6.6 shows the average number of occurrences of each error type in the solutions. In the initial solutions, the descriptive error type *Generalized Undo* and the conceptual error type *Stay Local* were the most common error types with a median frequency per solution of 5 and 7, respectively. The error types *Relaxed Car Unit*, *Early Unblock*, and *Avoid Blocking The Exit* occurred about three times in every solution. The error type *Equivalence Trap* was not observed in any solution and is not shown. When considering the revised solutions, it can be seen that almost all error types were eliminated. Only the error types *Stay Local* and *Relaxed Car Unit* are still present, on average (median) once in every solution. The error move classified as *Relaxed Car Unit* is actually a wrong move (and thus needs to be corrected by an additional move) and is exactly the move that prevented even the good participants from achieving the optimal solution of 25 moves (needing 27 moves instead).

For each error type, we additionally computed the number of the participants' solution attempts containing at least one occurrence of an error type. Table 6.5 shows that only five out of 115 participants' solutions in the initial attempts were free of any *Stay Local* error move. The error type *Relaxed Car Unit*—although not occurring in a high absolute number—was also found in almost all participants' initial solution attempts. But while the *Stay Local* error could be avoided by some participants in the revised solution (and was made by 77 participants in the revised solution), the error type *Relaxed Car Unit* was still contained in almost all participants' revised solutions. The finding that the number of solutions containing at least one *Relaxed Car Unit* error even increased from the initial to the revised solution can be explained as follows: There were several participants who did not solve the task in the initial solution and therefore did not reach a situation in which such an error is possible. In the second attempt, most of them were able to solve the task, so some of them also made a *Relaxed Car Unit* error. In general, we found that almost all error categories were present in the majority of the initial solutions.

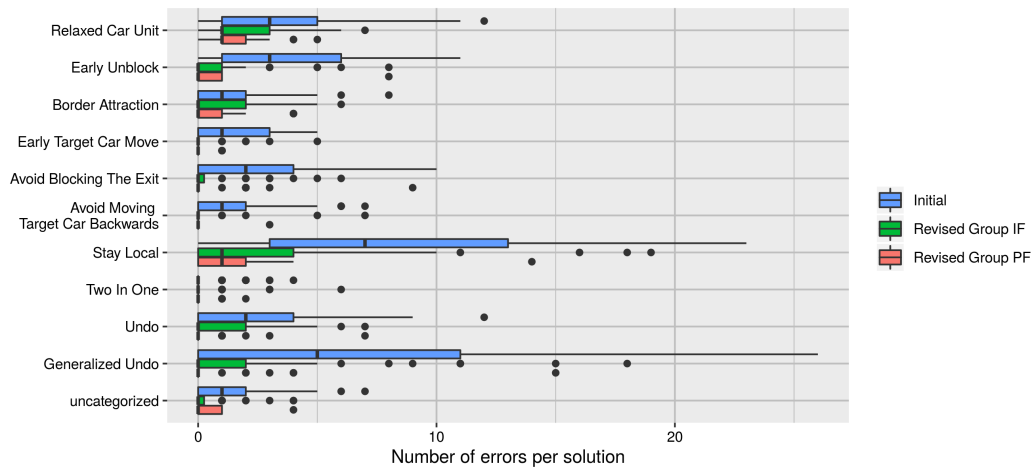


Figure 6.6 Experimental data: For each error category, it is shown how many error moves of this type are contained in the participants' solutions (initial and revised).

Table 6.5 For each error type, the number of participants who made at least one such error in their initial and revised solution is shown.

Category	Initial	Revised (both groups)	Revised (separate groups)
Relaxed Car Unit	106/115	110/115	52/56 IF 58/59 PF
Early Unblock	94/115	44/115	25/56 IF 19/59 PF
Border Attraction	66/115	44/115	26/56 IF 18/59 PF
Early Target Car Move	76/115	13/115	9/56 IF 4/59 PF
Avoid Blocking The Exit	77/115	25/115	14/56 IF 11/59 PF
Avoid Moving Target Car Backwards	59/115	7/115	6/56 IF 1/59 PF
Stay Local	110/115	77/115	41/56 IF 36/59 PF
Two In One	23/115	15/115	10/56 IF 5/59 PF
Undo	83/115	27/115	18/56 IF 9/59 PF
Generalized Undo	84/115	27/115	18/56 IF 9/59 PF

6.4.2 When in the solving process do the error types occur?

In addition to the pure frequency of occurrence of the error categories, we examined *when* in the solving process certain error types occurred. Since the solutions have different lengths (as the participants could also revisit the same situation twice), we did not consider the location of the errors relative to the number of steps of the solution. Instead, we considered the location of an error relative to the number of steps necessary to reach a goal state, i.e., the length of the shortest path from the error location to the closest goal state. This has the advantage that the location of errors in shorter and longer solutions can be compared. We understand that two game situations are not necessarily similar situations merely because their distance to a goal state is the same, but this approach is a sufficient approximation for this stage of the analysis. For the experimental level, the distance from the start configuration to the closest goal state was 25 steps, and there existed no reachable game configuration with a distance of more than 25 steps to a goal state. Figure 6.7 shows the number of errors dependent on their location of occurrence as distance to the goal state; the number is normalized by the number of participants (in the corresponding group).

It appears that there are two patterns of occurrences: Some error types only occurred in very specific game situations (at least at specific distances), while other error types occurred independent of the situation, but rather in the first half of the solution. Most conceptual error types are situation-dependent: For both *Avoid*-categories, for *Border Attraction*, and for *Relaxed Car Unit*, the peaks of occurrence are rather narrow. For the error categories *Stay Local*, *(Generalized) Undo*, and *Early Target Car Move*, the location of occurrence is not constrained to specific distances, but they mostly occurred in the first half of the solution. There are three possible explanations for this finding: (i) It is possible that the participants followed an exploratory “trial-and-error” approach in the beginning of the solving process, provoking a large number of *Stay Local* and *Undo* errors. In the further course of the solving process, this approach was replaced by a more target-oriented strategy due to learning effects in the solving process. A target-oriented strategy causes a decrease of the error types *Stay Local* and *Undo*. (ii) Another possible explanation is that planning the next move becomes easier when the goal to be reached is “within sight”. The smaller the distance to the goal state gets, the more probable it becomes that the participants will be able to plan the remaining moves needed to reach the goal. This might also explain the decrease of these error types in the second half of the solution. However, the decrease of error occurrences was found in the distance range of 10 to 15 steps to the goal state which—we assume—seems too long to account for a complete planning of the remaining steps. (iii) Since we only considered the number of error occurrences independent of the concrete game situations, it might also be possible that certain error types are only *possible* in specific situations—and that some participants made these errors only in these specific situations, while errors such as *Stay Local* or *(Generalized) Undo* are basically always possible. The question is thus whether there are errors that are made particularly often in relation to the number of situations in which it is actually possible to make them, while others are only made rarely in relation to their possible number of occurrences. In Section 6.4.3, we will examine this question.

When comparing the error occurrences in the initial and revised solutions, it can be observed that the number of those errors that rather occurred in the first half of the solution decreased considerably in the revised solution. This supports the hypothesis that these types of errors occur less often after the participants have spent a certain amount of time attempting to solve the task, due to learning effects. This is also supported by the observation that the decrease of these errors in the revised solution was also found for the group IF, which did not get any external material before the second solution attempt. Other error types also occurred less often in the revised solution. However, when we looked at the situation-dependent error types *Border Attraction* and *Relaxed Car Unit*, we found that their peaks of occurrence are almost of the same height for both the initial and the revised solution. For other situation-dependent error types, the peaks of occurrence are at the same distance, but the number of their occurrence dropped considerably from the initial to the revised solution.

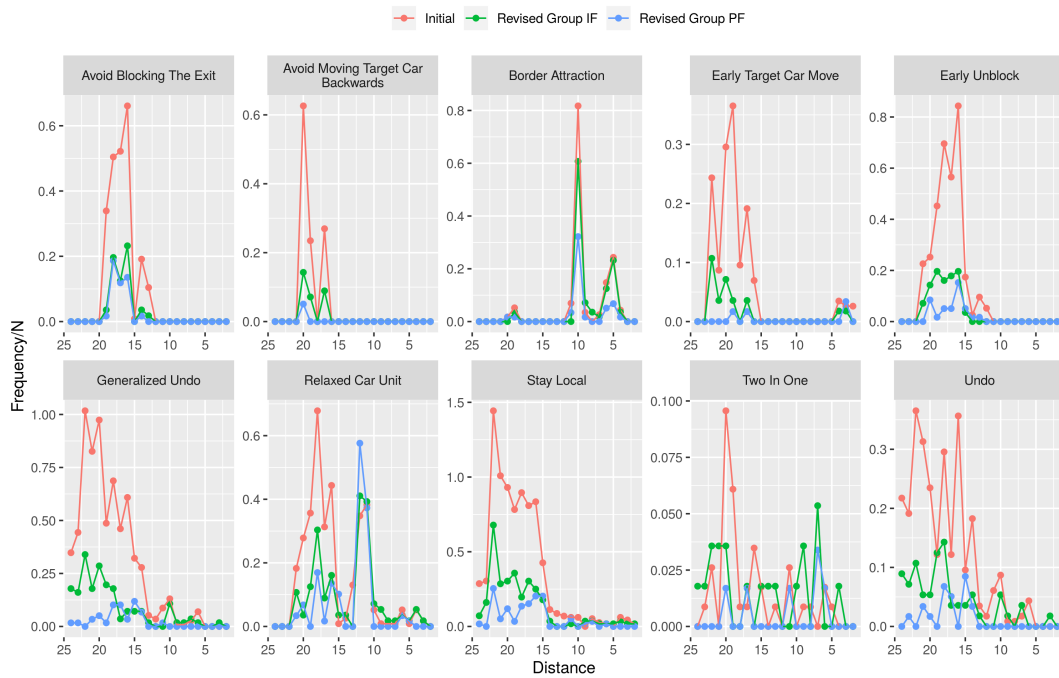


Figure 6.7 The number of error occurrences dependent on the distance to the goal state (on the x -axis) is shown separately for each error category. The absolute number of error occurrences is normalized by the number of participants (on the y -axis).

In order to investigate whether the observed narrow peaks of occurrence for certain error types are due to the fact that these error types are only possible in these situations, in the next section, we will compare the actual error occurrences to the possible number of error occurrences.

6.4.3 Which error types are often possible, but rarely made (and the other way around)?

The previous paragraph showed that some error types only occur at specific locations in the solution. This raises the question whether these errors occur only at these locations because they are only *possible* in these situations and not possible in others, or whether there are situations in the game that provoke certain error types – although they are also possible in other situations. To answer this question, we performed the following analysis: For each solution of each participant and for each visited game configuration, we checked which move alternatives the participants had in this situation and which alternative they chose. Each erroneous alternative was then categorized according to the error category system. For each state of the participant’s solution, we then know which types of error moves can be made here.

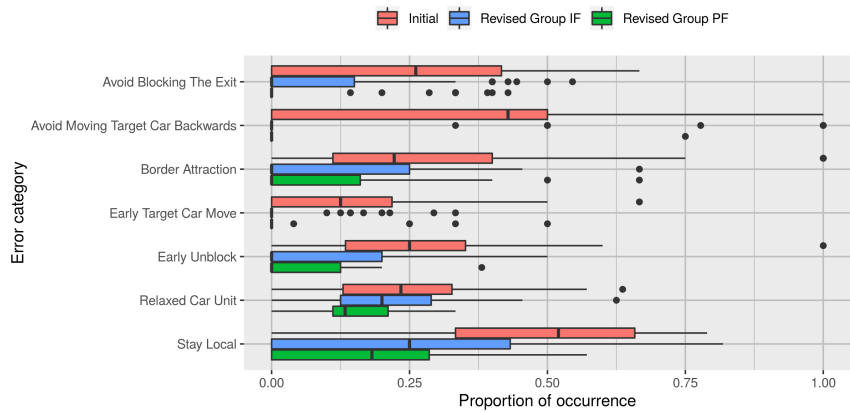
For those error categories that are dependent on the previous move of the participant, i.e., *Stay Local* or *Undo*, the previous state in the participant’s solution was considered. In the case of *Relaxed Car Unit*, the categorization might be dependent on the participant’s subsequent move which is obviously non-existent for an alternative move that was not made. Therefore, the following approach was taken: Consider a participant’s move (v, w) moving car j and all moves possible in configuration w , with successor states denoted as x_1, \dots, x_k . If any of the moves (w, x_l) with $l \in \{1, \dots, k\}$ moving car i is an error move and car j and i fulfill the requirements for a *Relaxed Car Unit*, move (w, x_l) is categorized as a *Relaxed Car Unit* Mistake. However, the idea of the *Relaxed Car Unit* Mistake is that a correct move with a car j provokes a mistake move with another car i because they

are perceived as a unit. Since they are considered as a unit, for the categorization of the mistake, it should not make a difference in which order the two cars are moved: whether the first move is correct and followed by the mistake or the other way around. For a participant's move (v, w) , we therefore check all sequences of two moves possible from w . If for such a sequence $w \rightarrow x_l \rightarrow y_l$, it holds that (w, x_l) is a mistake move, (x_l, y_l) is a correct move and cars of a car unit are moved, we say that a *Relaxed Car Unit Mistake* is possible in configuration w .

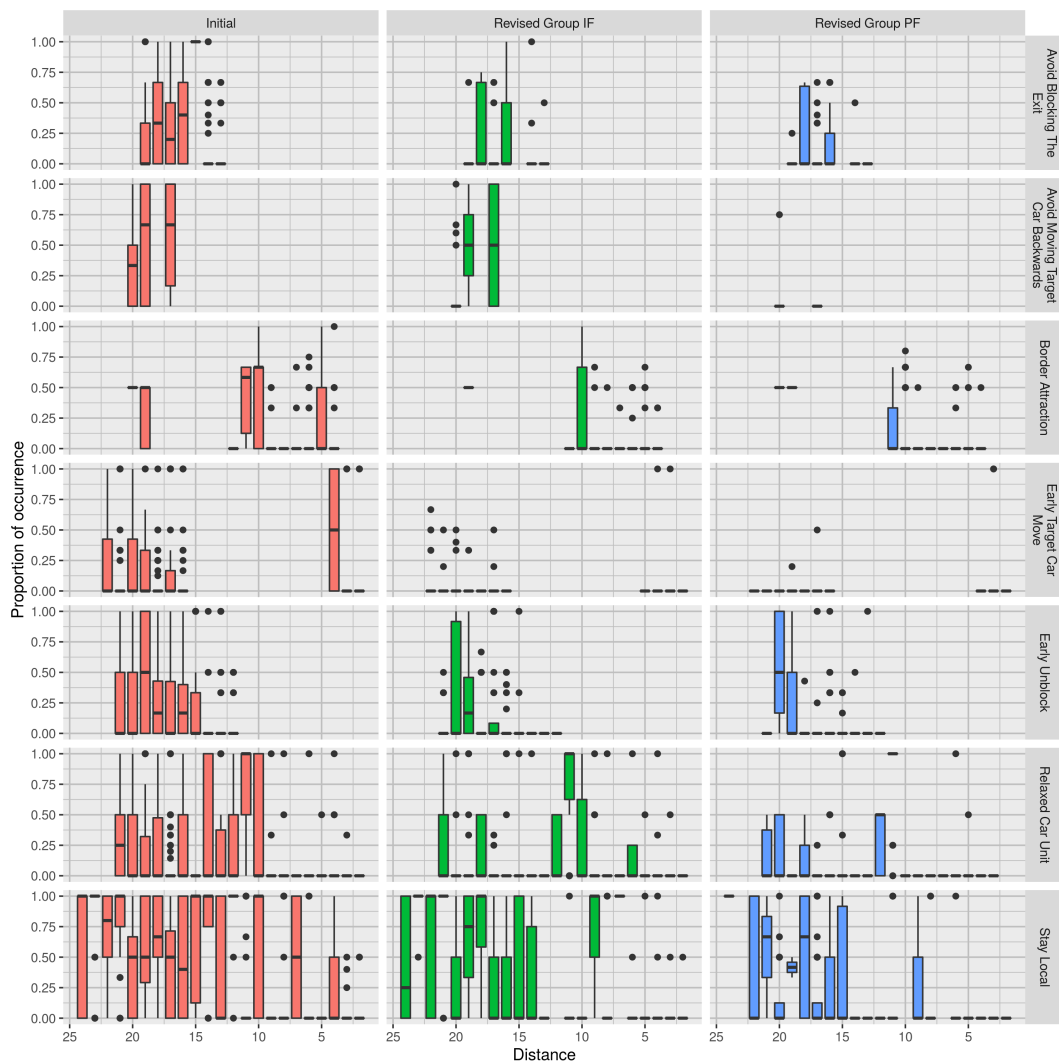
It is clear that descriptive error types are almost always possible; therefore, the following analysis focuses only on conceptual errors. For each participant's solution, each error type, and each distance from the goal state, we counted in how many states (contained in the participant's solution) the particular error type is possible and in how many states the participant actually chose a move of this error type. The relation between these two frequencies, called *Proportion of occurrence*, is shown in Fig. 6.8, once separately for each distance to the goal state (Figure 6.8b) and once aggregated over all distances (Figure 6.8a), in both cases aggregated for all participants in the PF-groups and the IF-groups, and separately for the initial and the revised solution. Figure 6.8a shows that in the initial phase, the errors of the types *Avoid Moving Target Car Backwards*, *Stay Local* and *Avoid Blocking The Exit* were made most often in relation to the number of situations in which they are possible. This decreased significantly in the revised solution. Nevertheless, although it occurred less often than in the initial solutions, *Stay Local* was the error type made most frequently in the revised solutions: It was still made in almost a quarter of all situations in which it is possible. Furthermore, the observation from the previous analysis could be confirmed: The proportion of occurrence of *Relaxed Car Unit Mistakes* did not change considerably from the initial to the revised solution. Figure 6.8b provides an explanation for the narrow peaks of the errors of the types *Relaxed Car Unit* or *Border Attraction* in Figure 6.7. For the type *Relaxed Car Unit* with a narrow peak at distance 11 and a wider peak at distance 21 to 16 in Figure 6.7, Figure 6.8b reveals that this error type is also possible at other distances, but was rarely or never made there. Only at distance 11, almost all participants (median proportion of occurrence of 1) perform an error move that is a *Relaxed Car Unit Mistake*. For other error types with distinct peaks of occurrence in Figure 6.7, Figure 6.8b shows that they are actually only possible at specific distances in the course of the game. For example, for the error type *Avoid Moving Target Car Backwards*, it can be seen that this error is only possible in the participants' solutions at distances 21, 20, and 19. But there (in the initial solution and the revised solution of group IF), this error was made in approximately 50 % of all cases.

6.4.4 Which errors could be avoided in the revised solution?

When comparing a participant's initial and revised solution, it is interesting to see whether the participant was able to avoid a specific error in the revised solution that they had made in their initial solution attempt—or whether in certain game situations, errors persisted also in the second attempt. This kind of question is particularly interesting in learning contexts of any task: For a teacher (or any automatic feedback system), it is important to know which kind of errors can be avoided by a learner in a second attempt without any intervention, and for which type of errors, an intervention by the teacher or the system is needed to help the learner. There are many errors made by learners when initially solving a task that they can detect by themselves—no intervention or external feedback is needed to help the learner to improve. Other errors, however, will persist also in a second attempt if there is no intervention or feedback for the learner from the teacher or the system. It is therefore of relevance to distinguish between these two types of errors. This leads to the question of which factors are necessary to enable a learner to avoid repeating mistakes, or—vice versa—which factors cause the learner to repeat an error. In our experimental setting, the previous paragraph showed that the majority of the error types made in the initial attempt could be avoided in the revised solution while others were still present in the second solution attempt. In order to recognize whether *the same* errors were made in the second attempt as in the first one, it is not sufficient to compare the frequency of occurrence of each error category. As already



(a) Proportion of occurrence of each error type, aggregated over all game states of all participants' solutions of the same group.



(b) Proportion of occurrence, separately for each distance to the goal. Distances for which no state is contained in the solutions in which an error type is possible at all are left blank.

Figure 6.8 For each game state contained in the participants' solutions, we computed which type of error moves would have been possible. Thus, for each error category, we could compute the proportion of occurrence, and the number of times that an error move of this type was made in relation to the number of times in which such an error type was possible. The upper plot aggregates these proportions of occurrences for all participants of each group, while the lower plot computes these proportions of occurrence separately for each distance to the goal state.

mentioned in the previous paragraph, there were participants who did not proceed far enough in solving the task in the initial attempt, so they simply could not make certain errors. It might be that a participant makes an error move in a certain situation in the first attempt and is able to avoid this error in the same situation in the second attempt, but by proceeding further in the second attempt, they are in new situations where they make the same error type (but in a different game situation). Furthermore, only because two errors in two solutions are categorized as the same type does not necessarily imply that they are conceptually identical.

We therefore need to define a notion of *similar* error moves in different solutions. This is not obvious because two error moves can be conceptually similar although they do not involve exactly the same game configurations. Consider, for example, the wrong move shown in Table 6.1 for the category *Border Attraction*. Assume that one of the two horizontal cars positioned under the target car is one cell further to the right than it is in the figure and that the pictured move is performed. Although these are not exactly the same game configurations, these moves should be considered as similar error moves.

We therefore introduce the following notation of *similar error moves*.

Definition 6.1: Similar error moves

An error move $m_i = (v_i, w_i)$ contained in one solution and an error move $m_r = (v_r, w_r)$ in another solution are said to be *similar* (noted as $m_i \simeq m_j$), if

- (i) m_i and m_r move the same car,
- (ii) in the same direction,
- (iii) m_i and m_r are either both unnecessary or both wrong moves, and
- (iv) if $d_F(v_i)$ and $d_F(v_r)$ denote the minimal number of steps necessary to reach a goal state from v_i and v_r , respectively, it holds that $|d_F(v_i) - d_F(v_r)| \leq 2$.

The last requirement assures that only moves in approximately similar game situations are called similar.

For three error moves $m_1 = (v_1, w_1)$, $m_2 = (v_2, w_2)$ and $m_3 = (v_3, w_3)$ which are all similar to each other, we say that m_1 is *more similar to m_2 than to m_3* if

$$|d_F(v_1) - d_F(v_2)| < |d_F(v_1) - d_F(v_3)|.$$

The threshold of 2 was determined specifically for this particular task with the intention of only considering moves as similar that are very close to each other. For a different game configuration, a different threshold might be appropriate.

We then performed the following matching procedure for the error moves of each participant: Let I be the participant's initial solution, R their revised solution and P the peer solution (the same for all participants). For each error move m_i of I , each error move m_p of P and m_r of R (if a participant's solution contains exactly the same error move twice, it is only considered once), we checked whether

- (i) $m_i \simeq m_r$ and $m_r \simeq m_p$ and $m_p \simeq m_i$ (**a perfect match**: a similar error move is contained in all three solutions), or
- (ii) $m_i \not\simeq m_r$ and $m_r \not\simeq m_p$ and $m_p \not\simeq m_i$ (**no match**: the three solutions do not have any similar error move in common), or
- (iii) $m_i \simeq m_r$ and $m_r \not\simeq m_p$ and $m_p \not\simeq m_i$ (**partial match**: there is a similar error move in initial and revised solution, but no corresponding error move in the peer solution), and the three corresponding combinations, or
- (iv) $m_i \simeq m_r$ and $m_r \simeq m_p$ and $m_p \not\simeq m_i$ (**partial match that needs to be resolved**) and the three corresponding combinations.

Table 6.6 The table shows how many error moves of the initial solutions, the peer solution and the revised solutions matched. Y/N denotes whether an error move was contained (Yes) or not contained (No) in the corresponding solution. The first letter represents the initial solution, the second letter the peer solution, the third letter the revised solution. There are for example 350 error moves (summed over all participants) where a similar error move is contained in both the initial and the revised solution, but not in the peer solution.

Perfect match YYY	Partial match			NNY	No match	
	YYN	NYY	YNY		NYN	YNN
152	219	78	350	179	551	1300

Partial matches that needed to be resolved were transformed into partial matches by matching only those two error moves out of the three that are more similar to each other and not matching the other one. If they were equally similar which was the case for very few move triples, the decision was made manually. Note that an error move of any solution can be contained in several matches (this is the case for only very few error moves), for example if an error move in the revised solution was similar to several error moves in the initial solution. But it is not possible for an error move to be contained in more than one different type of matches: It is either contained in perfect matches, or in partial matches, or in no match.

Table 6.6 shows the number of perfect, partial and no matches obtained by this procedure. We use the following short notation for matches, consisting of three letters: The first letter corresponds to the participants' initial solution, the second one to the peer solution, the third one to the revised solution. Yes or No indicate whether a corresponding similar error move exists for a match: YYN for example contains partial matches where a similar error move is contained in a participant's initial solution and in the peer solution ($m_i \simeq m_p$), but no corresponding error move is contained in the participant's revised solution ($m_p \not\equiv m_r$ and $m_i \not\equiv m_r$).

Although the category of the error moves was not considered in the matching procedure, the error moves matched by this procedure do fit very well with respect to their categorizations: For all matched error moves of the initial and the revised solution, 84% have exactly the same conceptual error categories, while 13% differ in exactly one category. These proportions are similar for the matched error moves of the initial and the peer solution (85% and 12%) and for the matched error moves of the peer and the revised solution (98% and 2%).

The procedure of matching error moves allows analyzing which error moves were repeated and which ones could be avoided by a participant in the revised solution. Table 6.6 shows that the vast majority of error move matches are no matches, mostly YNN, i.e., an error move is contained in a participant's initial solution, but there is no corresponding error move either in the peer solution or in the participant's revised solution. Among the perfect and partial matches, the pure frequencies shown in Table 6.6 indicate that it is very rare that an error *not* made in the initial solution, but contained in the peer solution, is then contained in the revised solution (case NYY). This would correspond to the situation that a participant unintentionally "learned" from the peer solution by adopting an error from the peer move that they had not made in the initial solution. However, this can also include scenarios in which a participant did not manage to solve the task in the initial attempt (and therefore simply could not make certain errors), and then used the provided peer solution as a solution draft together with all of its errors. A more common case is that an error contained in the initial solution as well as in the peer solution could be avoided in the revised solution (case YYN). In this case, it is possible that the participant benefited from generating feedback for the peer solution since the peer solution contained a similar error move as their own initial solution and detecting this error in another solution helped to avoid the same error in the revised solution. However, these interpretations cannot be derived from Table 6.6, but are rather speculations at this point, for the following reasons: First, Table 6.6 contains the number of error move matches

aggregated over all experimental groups. Half of the participants, however, generated feedback on the peer solution (and therefore *saw* the peer solution) *after* attempting to solve the task for the second time. Therefore, error moves in the peer solution could not have had any impact on the error moves in their revised solution. Second, it can be expected that it makes a difference whether a participant *noticed* that a move in their initial solution or in the peer solution was not correct. It can be expected that if a participant noticed that a move in their initial solution was not correct, they would not repeat the same error in the revised solution—*independent* of whether the error move was contained in the peer solution.

These considerations lead us back to the original question of this section: Which factors are actually essential to enable a participant to avoid an error in the revised solution and which factors lead to a repetition of the same error?

To answer this question, we used additional data collected in the experiment: Every participant generated internal feedback on their own initial solution and feedback on the peer solution. It is therefore known for each error move in the initial solution whether they recognized it as a mistake directly after the solving process. The PF-groups also generated feedback on the peer solution, which is why for the participants of those groups, it is also known which of the peer error moves were recognized as mistakes⁴.

We used this data to generate a decision tree that provides predictions on whether an error will be repeated or avoided in the revised solution. In general, decision trees are used as simple predictors for the class membership of a set of elements [Qui86, RM08]. If each element is associated with a set of features, e.g., represented by a vector \vec{x} and a class variable, a decision tree trained with a set of elements for which the class membership is known (test data) is a model that is able to predict—based on the vector \vec{x} —the class of new elements. This is achieved through the generation of a binary tree where each leaf node is associated with a class, each internal node represents one feature and the outgoing edges of internal nodes correspond to possible values of the feature. If the class for a new element is to be predicted, the tree is traversed from top to bottom. For each internal node, the value of the corresponding feature is checked; and the corresponding tree branch is used. This procedure is repeated until a leaf node is reached and the prediction for the class of the new element is according to the class associated with the leaf node.

Standard methods for generating such a decision tree usually work top-down: From the root to the leaves, at each step, that feature is chosen as the current node that best separates the given data where “best separation” can be measured by several metrics. Since trees built in such a way are usually overly complex, the tree is reduced (“pruned”) in a second step [BFSO98].

For the data at hand, all matched error moves of all participants in their initial and revised solutions as well as in the peer solution correspond to the set of elements. The class variable to be predicted is the flag whether the error move is contained in the revised solution or not where R.Y means that the error move is contained in the revised solution, and R.N means that the error move is not contained in the revised solution. The following features were used:

Group $\in \{\mathbf{IF}, \mathbf{PF}\}$ Group association of the participant.

InI $\in \{0, 1\}$ Flag whether error move is contained in the initial solution.

InP $\in \{0, 1, \mathbf{NA}\}$ Flag whether error move is contained in the peer solution. Variable is set to *NA* if the participant is in group IF since participants of this group did not see the peer solution before the second solution attempt.

MAW_IF $\in \{0, 1, \mathbf{NA}\}$ Flag whether error move in the initial solution was marked as wrong by

⁴Participants in the IF groups also generated feedback on the peer solution, but *after* the second solution attempt. Therefore, for those participants, it is also known which error moves of the peer solution were recognized as error moves. But since this should not have any effect on the second solving performance, we did not consider this information for the question of which errors were repeated in the revised solution.

the participant in the internal feedback phase. If the error move was not contained in initial solution, MAW_IF was set to NA .

$MAW_PF \in \{0, 1, NA\}$ Flag whether the error move in the peer solution was marked as wrong by the participant in the peer feedback generation phase. If the error move was not contained in the peer solution or the participant was in the experimental group IF, the flag was set to NA .

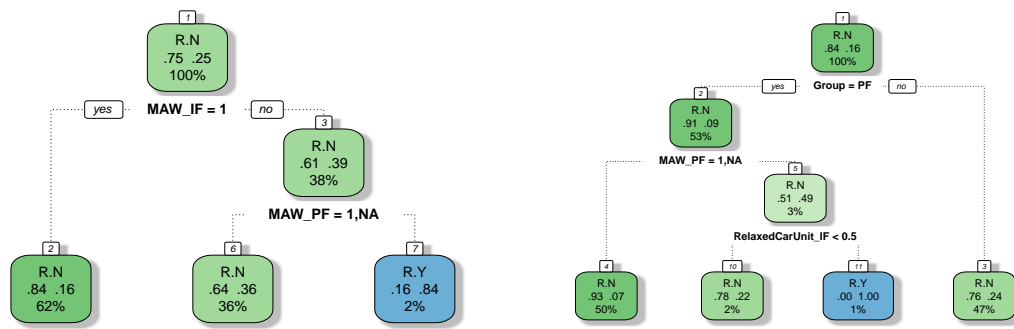
Error categories For each error category and each solution type (initial, peer, and revised), there is a flag indicating whether the error move in the solution was categorized as such an error. If the error move in the initial solution was categorized as *Relaxed Car Unit*, the flag $RelaxedCarUnit_IF$ is 1, if it was not categorized as such an error, the flag was set to 0. If the error move was not contained in the initial solution, the flag is set to NA . Since the error categorization of the matched error moves coincided for almost all matched error moves, it is sufficient to use the categorizations of the error moves of only one phase.

Figure 6.9b shows a graphical representation of the generated decision trees. Below each node, the feature name and condition are shown by which the data is split at this node. The left branch is applied on the data entries for which the feature condition is true, the right branch for data entries for which the feature condition is false. In each node, the first line represents the prediction for the data entries for which this node is valid, the second line represents the proportion of each class in the sub-data where the left value is, in this case, the proportion of error moves not contained in the revised solution (R.N). The third line indicates to which proportion of the total data the node applies.

Which factors predict the repetition of an error move in the revised solution? In order to investigate which factors are essential so that an error is not repeated in the second solution attempt, we constructed a decision tree based on those error moves contained in the initial solution (i.e., the matchings starting with the letter *Y*). Figure 6.9a shows the corresponding decision tree. It can be seen that the feature discriminating best between the repetition and the avoidance of an error move is the factor MAW_IF , i.e., whether the error move was marked as wrong by the participant. This applies to 62% of all matched error moves. It is, however, astonishing that 16% of these error moves—marked as wrong by the participant—were repeated in the revised solution. For those error moves where the participant did not notice that it is not a correct move, the second discriminating feature contained in the decision tree, is MAW_PF , i.e., whether the corresponding error move was detected as wrong in the peer solution: If a participant did not realize that a move was not correct, they could benefit from seeing another person's solution attempt. If they were able to realize that the corresponding error move in the peer solution was wrong, a repetition of the error in the second solution attempt becomes less probable. Although this is an intuitive result, it needs to be noted that the depicted decision tree does not offer a perfect prediction for the error moves at hand: For 23% of the error moves, the decision tree yields a wrong prediction about the repetition of an error move, although these are the error moves from which the decision tree was generated.

Which factors predict the repetition of an error even though it was detected as an error?

The decision tree shown in Figure 6.9a reveals that 16% of these error moves contained in the initial solution *and* marked as wrong by the participant were repeated in the revised solution. This is an unexpectedly high number since we would expect that realizing that a move was wrong should enable the participant to avoid the same error move in a second attempt. In order to explain this high number, we generated a second decision tree for predicting the occurrence of an error move in the revised solution, which is now based on data that includes only those matched error moves that are contained in the initial solution *and* were marked as wrong by the participant (a total of 1256 error moves). Besides the participant's group, the discriminating features were MAW_PF , i.e., whether the participant was able to detect the corresponding error in the peer solution, and the flag showing whether the error move was categorized as a *Relaxed Car Unit* error. If the error was



(a) Only for those error moves which occur in the initial solution.

(b) Only for those error moves which occur in the initial solution and were marked as wrong by the participants in their own solution.

Figure 6.9 Decision trees predicting whether an error move will be repeated by a participant in the revised solution (R.Y–error is contained in revised solution; R.N–error is not contained in revised solution), dependent on the following factors: experimental group of the participant (PF and IF); whether error move was marked as wrong by the participant in internal feedback phase (MAW_IF=1–marked as wrong; MAW_IF=0–not marked as wrong); whether this error move is contained in the peer solution (InP=1–contained in peer solution; InP=0–not contained in peer solution); whether error move was marked as wrong by the participant in peer solution (MAW_PF=1–marked as wrong; MAW_PF=0–not marked as wrong); error category of error move in initial/revised/peer solution (e.g., RelaxedCarUnit_{PF,IF}=1–error move is categorized as *Relaxed Car Unit* in peer/initial solution; RelaxedCarUnit_{PF,IF}=0–error move is not categorized as *Relaxed Car Unit* in peer/initial solution).

contained in the peer solution and not marked as an error, and if it was categorized as a *Relaxed Car Unit* error, the repetition of the error in the revised solution can be predicted perfectly. Thus, it seems that perceiving two cars as a unit is sometimes such a strong incentive for moving *both* of them consecutively that such errors are even made if the participant is aware that this move is not correct.

6.5 Summary

In this chapter, the methods and results of an interdisciplinary research project were presented. While the research question—the impact of generating feedback on a learner’s performance—stems from the field of psychology, methods and approaches from computer science helped to analyze the experimental data and gain insights about the cognitive processes of human problem-solving. Due to the experimental design, the experimental data contained (among other information) two solution attempts of a Rush Hour task for each participant. The challenge was thus to compare the solution attempts of each participant and to evaluate the quality of each solution. While the number of errors in a solution is a simple and meaningful metric for quantifying the quality of a solution, it is clear that this metric is not sufficient to understand the problem-solving process. In order to better understand *why* one solution is worse than another, our approach was to focus on the error moves: Errors can occur for different reasons. We therefore aimed at providing a category system capable of automatically classifying error moves according to their (assumed) motive. While the proposed categories were formulated for the experimental task, a Rush Hour game, all categories are based on general principles that are also valid for other types of tasks.

The error category system allows a more detailed analysis of the experimental data. When comparing each participant’s first and second solution attempts, we found that most errors categorized as descriptive error types could be avoided by the participants in the second attempt. There were,

however, error types that were still present in almost all participants' second solution attempts. When considering at which points of time in the course of the game these errors occurred, we find two different patterns: Some error types occurred at specific situations in the game while others were situation-independent, but tended to only occur in the first half of the initial solutions.

Since one possible explanation for these patterns is that errors are only *possible* in certain situations, we considered for each move all of its alternative moves and analyzed which error types were possible in the game situations. For error types such as both *Avoid*-categories as well as for the error category *Early Target Car Move*, we could confirm that these errors are only possible in specific game situations. Other error types such as *Relaxed Car Unit* are almost always possible, but nevertheless, they were made only in specific situations—and then by almost all the participants. This indicates that certain game situations are more error-prone than others.

For learning contexts, it is often of relevance to know which factors are essential for a learner to be able to avoid an error—and which factors cause a learner to repeat an error. In order to answer this question for the experimental data, a notion of “the same error” in different solutions is required. We therefore proposed a method of matching the error moves of different solutions which allows identifying whether a participant's initial and revised solution and the peer solution contain the “same” error move. Although in most cases, an error made in the initial attempt was not contained anymore in the revised solution, there were a considerable number of cases in which the same error was repeated in the revised solution. In order to understand which factors explain the repetition of errors, we constructed decision trees predicting the occurrence of an error in the revised solution, based on its occurrence in the initial and the peer solution, its error category, and on whether the error was detected by the participant in the initial or the peer solution. We found that realizing that a move was not correct was the most important factor for not repeating an error. While this result is not surprising, we also found that 16% of all error moves contained in the initial solution and detected as error moves by the participants were still repeated in the revised solution. In these cases, the participant could have benefited from seeing another person's solution and could have detected the same error in the other solution. Interestingly, there is an error move categorized as *Relaxed Car Unit* which was repeated by most participants in the revised solution *although* they detected it as wrong in their own solutions.

We are convinced that although the proposed error category system was developed for solutions of Rush Hour tasks and the analysis was performed for these tasks, the methods can be transferred and applied to other well-defined sequential tasks. In general, we believe that there is great potential in interdisciplinary collaborations such as the presented project. Bringing together ideas and methods from different disciplines, can yield interesting results overall, and help to understand the cognitive processes of human problem-solving in particular.

Towards a process-driven network analysis

7.1 The relevance of network processes for network analysis

In Chapter 1, we named two aspects as motivation for this thesis which we think are both relevant for network analysis: (i) considering the relevant network process helps to determine appropriate network representations and measures; and (ii) considering empirical network flow in addition to the network representation offers valuable information that facilitates understanding the complex system. In the following, the results of this thesis will be contrasted with these two aspects of motivation.

Why taking into account the process is beneficial for network analytic projects It has been argued that in a network analytic project in which a real-world complex system is under investigation, the network representation, and the network methods cannot be chosen independent of the network process of interest [DLZ12]: First, the network representation and the network process need to match, since the network representation needs to be chosen such that the network edges represent the type of relations that are essential for passing on the process of interest. Investigating how an infection such as HIV will spread by using an online social network like Facebook will lead to results that are difficult to interpret. Second, the applicable network measures are restricted by the process of interest: Borgatti argued for common centrality measures, saying that each centrality measure contains a set of assumptions about a network flow [Bor05]. A centrality value then reflects a node's importance with respect to a network flow with those assumed properties. Applying a network measure with the assumption that a network flow uses shortest paths while one is interested in the process of information spreading, is technically possible, but the results are also difficult to interpret.

At the same time, when considering a real-world complex system, there are often dozens of different plausible network representations for the same set of system observations, and even more (technically) applicable network methods. It has been shown that seemingly minor decisions in network construction will lead to considerably different analysis results [But09, DCMHW10, TZ16]. Thus, the appropriate network representation for the system under investigation and the specific research question need to be chosen with care.

We therefore argue that—in cases where the research project is tied to a network process—a network analytic project might benefit from focusing on the relevant process in an early stage of the project. When it has been determined whether and which network process is relevant for the research question, the number of possible plausible network representations and applicable network measures can often be reduced. Therefore, a focus on the relevant network process helps to determine an appropriate network representation and applicable network measures.

Focusing on the relevant network process can also help to identify situations in which a network representation is not necessary at all: A network representation is a helpful representation when indirect effects are of interest. This form of representation focuses on the interactions between the entities instead of on the properties of the single entities. It allows analyzing effects that can only be explained by the connectedness of the entities, and not by the behavior of the single entities. This means, in particular, that entities can have an effect on each other even when they are not connected directly, but rather indirectly via intermediate links. Indirect effects can be caused by a network process flowing through the network: Pieces of information, infections, but also flows of passengers can cause indirect effects. Note, however, that this is not the only explanation. Consider for example a relationship such as similarity (of any kind) between entities. If such a relationship is transformed into a network representation where an edge is inserted if the two entities are sufficiently similar to each other, a network representation can, for example, be used to identify groups of similar entities. At the same time, there is no process flowing in this network. On the other hand, if network measures associated with a network process, such as centrality measures, are applied to this type of network, the interpretation of the results needs to be done with care.

Why empirical flow data should be considered In this work, we have shown that empirical network flows show different properties than simple process models. This is particularly interesting because commonly used network measures such as centrality measures, also measure the nodes' importance with respect to a simple process model. We have shown that incorporating the empirical flow data into these measures affects the nodes' centrality scores. At the same time, it is often the case that network representations are created by using trajectory data: A set of trajectories is aggregated into a network representation by inserting an edge from a to b into the network if at least x trajectories contain a transfer from a to b . This was also done in this work, for example to create the air transportation network from passenger journeys used in Chapters 3 and 4. Aggregating trajectory data like this is a convenient way to obtain a network representation of the infrastructure in which the trajectories take place. There are, however, mainly two drawbacks: First, by aggregating the trajectories, any dependencies contained in them, cannot be represented anymore. Consider the trajectories (x, y, z) and (x', y, z') where, for some reason, transitions from y to z are only viable if the previous transition is (x, y) . Aggregating all trajectories into one network representation cannot describe such dependencies anymore. In order to overcome this limitation, it has been proposed to use network representations of higher order [XWC16], where the nodes do not represent single entities anymore, but tuples of entities. Then, depending on the order k , dependencies of depth k can be represented. It is, however, not trivial to interpret results from network measures applied to a network representation of higher order. A second drawback of aggregating trajectories into a network representation is that any information about usage frequencies is lost: Possibly tens of thousands of trajectories are aggregated into one network representation where each edge has the same "quality"—independent of whether it is contained in one or in several thousands of trajectories. There are several options for weakening this effect, for example by introducing a threshold x such that an edge is only inserted if the corresponding connection is contained in at least x trajectories, or by introducing edge weights representing the number of trajectories that contain this connection. However, none of the options is a satisfactory solution: Introducing a threshold only removes the less used edges; and applying a network measure to a weighted network always needs to be done with care, since weights can carry different semantic meanings—and for different network measures adapted to weighted networks, the weights might be interpreted differently.

For these reasons, we are convinced that considering the empirical trajectories of network flows *in addition* to the network representation is a valuable approach in order to understand the properties of a complex system. Only recently have datasets containing such trajectories become available. In a manner of speaking, this situation is somehow similar to the early days of "complex network analysis" in the late 1990s when datasets of real-world networks became available, and—due to the increasing capacities of computers—their analysis became feasible. Similarly, when datasets of network flows now become available, methods and tools for analyzing such types of data are needed, which will be described in the following section.

7.2 Thinking about network analysis from a process perspective

Complex network analysis comprises several aspects for which an analogy for network processes can be drawn:

Structural properties When we consider network representations of real-world systems, it turned out that most network structures found in real-world systems showed similar structural properties although they were derived from different contexts. Many real-world networks share the property of scale-freeness, meaning that their degree distribution follows a power law [BA99] (this finding has, however, been questioned recently [BC19]). For processes in networks, there is an analogy: In Chapter 3, we showed that real-world processes also have such characteristics. There are a few nodes and node pairs that are used very often by the process while most are only used a few times or not at all. Future work needs to show whether there are more common or distinguishing properties of real-world network flows.

Methods as analytical tools There exist many methods that are used as analytical tools in order to gain insights about the underlying complex system: Centrality measures are designed to identify the most important actor in the system [KLP⁺05], clustering approaches to find groups of actors in the system, link prediction mechanisms to predict the future evolution of the system, etc. Similarly, network processes can also be used to infer knowledge about the system: While existing network measures only use the structure of the network, methods incorporating the actual usage of the network by the process might be able to provide better insights about the underlying system. Possible application scenarios that are beyond simple usage frequencies, are flow-based centrality measures as proposed in Chapter 4, or definitions of node equivalences where a similar usage pattern determines nodes with the same functionality, or flow-based methods of community detection.

Models When analyzing the structural properties of a network representation, an appropriate comparison is needed to determine whether the identified properties are remarkable in any sense. This is usually done by comparing the structural properties of the real-world network with a suitable network model created artificially. Another important use case for network models is their explanatory power: When a model created with only a few simple rules is able to reproduce the structural properties of a real-world network, this leads to the presumption that these simple rules also played a role in the formation of the real-world network. Due to these two reasons, several classes of network models have been proposed. In the early days of network analysis, there were basically two types of models for graphs: regular graphs, i.e., graphs with a lattice structure, and random graphs, i.e., graphs with randomly created edges. However, it turned out that the structure of real-world networks can be explained neither by random graphs nor by regular graphs. While regular graphs exhibit a high clustering coefficient and large characteristic path lengths, random graphs show small clustering coefficients and short characteristic path lengths. Real-world networks, however, often have a high clustering coefficient and small characteristic path lengths at the same time. Watts and Strogatz then introduced their small-world network model [WS98] which is able to scale between those two extremes with a parameter p .

For a certain range of p , the created networks show a large clustering coefficient (like real-world networks and regular graphs) *and* small characteristic path lengths (like real-world networks and random graphs). There is a similar situation regarding network processes: There are mainly two types of models for processes: models that move on shortest paths and models that move randomly through the graph (in many different variants). In Chapter 3, we showed that the real-world network flows we considered neither moved on shortest paths nor totally randomly, but fell somewhere between these extremes, which is in analogy to Watts' and Strogatz' small-world model. We

Table 7.1 Aspects of classic network science with an example and how they can be transferred to a process-based perspective.

Aspect	Example from network analysis	Transfer to process perspective
Structural properties	Degree distribution: power law [BA99]	Source-target distribution: power law? (Chapter 3)
Methods	Centrality measures[BE06]: use network structure	Flow-based measures: use network usage (Chapter 4)
Models	Small-world model: between regular and random graphs [WS98]	Trajectory model: between shortest paths and random walks (Chapter 3)
Data preprocessing	Impact of data collection and decisions in network creation [But09]	Impact of data collection and decisions in trajectory creation

compared the real-world flow trajectories to a suitable set of shortest paths and random walks and showed that the real-world flow trajectories share certain properties with both shortest paths and random walks. A suitable model for real-world flow trajectories is thus needed.

Dimensions of data preprocessing An often neglected aspect of network analysis is the procedure of transforming the observations of the system into a network structure. It has been shown that the results of any network analytic method are highly dependent on the researcher's decisions in the data preprocessing phase [But09, DCMHW10]. Decisions that a researcher might make to construct a network representation from a system are for example:

- Which observations of the system are included: Restrict to which time window of observations? Filter out seemingly erroneous observations?
- What entities are represented by a node: Aggregate entities [But09]? Which entities to include [LMP83]?
- What interactions are represented by edges: Introduce a threshold [DCMHW10]? Aggregate over time?

These considerations and their implications also need to be made for the analysis of processes: When observing a process in the system, it is not always obvious how to extract single process trajectories from these observations. Future work needs to analyze how robust the analysis of network processes is against the modeling decisions in the data collection and preprocessing phase.

In recent decades, network analysis has been proven to be a useful tool for understanding complex systems in various disciplines. In this approach, the system is represented as a network in which the nodes represent the system's entities and the edges represent the entities' interactions. Thus, the focus is on the *interactions* within the system instead of on individual analyses of the single entities. Nowadays, there is a large number of pre-compiled datasets containing a network representation of a system, and easy-to-use software libraries that enable analyzing these network structures. However, it is sometimes overlooked that a network representation and all network analytic methods—like almost all representations and methods—include a set of assumptions: A network is a convenient representation if indirect effects in the system are of interest. If such indirect effects are not present, a different form of representation or different analysis tools might be suited better. Also each network measure contains assumptions. For centrality measures, Borgatti [Bor05] pointed out that all classic centrality measures assume the presence of a network process, i.e., “something” is flowing through the network from node to node. A centrality measure can then determine the importance of a node with respect to such a flow. These processes can also explain indirect effects in a system: A can have an impact on C via B if, for example, a piece of information is transferred from A via B to C. Centrality measures that assume the *presence* of a process, also assume certain properties of this process [Bor05]. Borgatti notes that the most frequently used centrality measures assume a process using only shortest paths or a process being spread by a parallel duplication mechanism. However, many processes for which a centrality measure would be relevant, do not exhibit these properties.

To make matters worse, even for network processes where these assumptions seem justified, our analysis in Chapter 3 showed that the assumptions are also not met when considering datasets of real-world network flows. Neither of the considered datasets showed exclusive usage of shortest paths. Other assumptions of classic centrality measures were found not to be met, either. For all datasets, we found node pairs between which there was a large proportion of the total amount of flow while there was only a small amount of flow (or no flow at all) between most node pairs. Our finding was similar for the assumption that the flow is distributed equally among equivalent path alternatives. Here, we found that the real-world network flows rather preferred one alternative instead of equal distribution among the alternatives, although how strong this effect was varied among the datasets we considered.

A naturally arising question from these findings is: Which impact does it have on existing centrality measures that their assumptions are not completely fulfilled? In order to answer this question, we introduced flow-based variants of closeness and betweenness centrality in Chapter 4. In each variant, either the original measure's assumption was used for the computation or the actual behavior of the real-world network flow. One betweenness variant, for example, counts the number of real trajectories in which a node is contained instead of the number of shortest paths. Another betweenness variant considers only those node pairs for the computation between which there is flow in the real-world network flow, instead of all node pairs. A similar approach was presented for closeness centrality. This approach allows a systematic evaluation of the measures' assumptions and an answer to the question: Which impact does the violation of an assumption have on

the results of a centrality measure? We found that for all considered datasets, the standard centrality measures showed a high correlation to the flow-based variants indicating a robustness of the measures against violations of their assumptions. At the same time, for all datasets, we found nodes whose ranking position deviated considerably in at least one flow-based variant compared to the standard centrality measure. Also nodes ranked high by the standard centrality measure were concerned. This has consequences for the application of the standard centrality measures in practice: When applying a centrality measure to a network, it is usually only the highest-ranked nodes that are of interest. We found that even those nodes can be affected by taking into account the actual flow dynamic in the network: Nodes ranked highly by the standard centrality measure decreased in importance when the real-world flow was considered—and nodes that were important with respect to the actual network flow were not ranked high by the standard centrality measure. Our results in Chapter 4 thus show that rankings of centrality measures need to be interpreted with care.

Our results also show the importance of taking into account the flow dynamic on the network when analyzing a network: In some cases, a static network representation without any information about the flow might simply not be sufficient for understanding the system. In these cases, reducing the system to a network structure and applying measures to the *structure* of the network might be an oversimplification of the system. This is particularly the case when flow data is used to generate a network as in the case for the air transportation data described in Chapter 3. Here, the flow trajectories are used to infer the network structure and in this inference, a lot of information is lost. When network analytic methods are then applied to the network representation, the results might be misleading. An illustrative example of such a case is given in Chapter 4 for the US air transportation system: When building a network structure from the passenger journeys and applying betweenness centrality to this network, the Alaskan airport Anchorage is among the highest-ranked nodes although only a small proportion of the network flow involves this airport. This effect is due to the fact that there are many airports in Alaska, but all of them are small, process only a negligible amount of passenger flow, and almost none of them is connected to any airport outside Alaska—except Anchorage, which therefore serves as a gateway between the contiguous United States and the state of Alaska. Thus, if we only consider the network structure in which an edge between two Alaskan airports and an edge between New York City and Washington, D.C. is of the same quality, the betweenness centrality does what it should do—perfectly: The node Anchorage is identified as a gateway node and therefore ranked high. If, however, we consider the actual passenger flows, it becomes obvious that the node Anchorage should not be the most central node of the network. We therefore argue that using the actual network flow—if available—is beneficial for a more profound analysis.

When handling real-world datasets of network flows consisting of a large number of trajectories, computational resources might limit the analysis of such datasets. Therefore, a method for summarizing the set of trajectories into groups is beneficial, since then, a subsequent analysis can be performed on the set of representative trajectories of each group. To achieve this goal, a method for finding groups of “similar” trajectories is needed. It is, however, not clear how the similarity of two trajectories can be measured and which similarity measure is to be preferred over another. In Chapter 5, we therefore presented a systematic evaluation of existing distance and similarity measures, adopted on trajectories. Similarity and distance measures from different domains, not necessarily designed for trajectories, were introduced and their ability to find meaningful groups was evaluated by using a dataset with ground truth. We found that distance measures based on the discrete Fréchet distance yielded groups of trajectories with a clearly higher purity with respect to the ground truth than other distance and similarity measures. We postulate that this result is due to the fact that these distance measures take into account a trajectory’s set of nodes, their order, and their position in the network, while other similarity and distance measures only take into account one of these aspects.

As a case study in which a set of trajectories in a network was used to gain insights about the

domain itself, we presented the results of an interdisciplinary project in Chapter 6. Together with researchers from the department of Psychology at TU Dresden, we conducted an experiment in which the participants were asked to solve a planning task, i.e., to solve a medium level of a single-player board game. By understanding the participants' solution drafts as trajectories in the game's state space, new and more sophisticated methods of analysis become possible. In order to understand the difficulties in solving the given task and to evaluate the participants' improvement in their second attempt, we proposed an error category system: A rule-based system labels each error move contained in a participant's solution attempt according to the assumed motive for why this error move occurred. Although the proposed category system was formulated for the specific task used in the experiment, the categories are based on general cognitive principles, which can be transferred to other types of tasks as well.

In general, this thesis demonstrates the relevance of considering *network processes*:

- (i) A focus on the network process relevant for a specific research question in an early stage of a network analytic project will help to reduce the number of appropriate network representations and the number of applicable network measures.
- (ii) The analysis of datasets containing real-world network flows has shown that real-world network flows show different properties than simple, commonly used flow models. Thus, in order to analyze networks with respect to the *actual* flow, real-world data needs to be taken into account.
- (iii) This is particularly true for network measures assuming a network flow, such as centrality measures. The results presented in this thesis showed that commonly used centrality measures overestimate and underestimate the importance of nodes with respect to the *actual* network flow. Thus, when applying a centrality measure on a network, its results need to be interpreted with care.

8.1 Future work

The work presented in this thesis can serve as a starting point for further research dedicated to the analysis of network flows, particularly with respect to existing network measures and methods. We therefore see several aspects for future work:

- A major challenge of this work was the collection of suitable datasets containing real-world network flows, since only a few suitable datasets are available. A big task for the whole research community is therefore to collect of further datasets, and make them accessible to the community.
- This work mainly focused on centrality measures and reviewed this family of network measures with respect to their assumptions. In future work, other types of network measures also need to be evaluated with respect to the assumptions incorporated in them, particularly with respect to assumptions about process properties. Using the available network flow datasets, those assumptions can be evaluated as we did in this thesis for centrality measures.
- In Chapter 3, we compared the properties of real-world network flows to the two simplest models for trajectories, shortest paths and random walks. Since both extremes did not yield satisfactory results such that one of these models was able to reproduce the properties of real-world network flows, future work can be dedicated to investigate whether more sophisticated trajectory models are able to yield better results. While there exist dedicated trajectory models for trajectories of specific domains, a unifying model applicable for trajectories from different domains is missing—like, for example, the Small-World-model, which is valid for networks from different contexts.
- In Chapter 4, flow-based variants of centrality measures were introduced for testing the impact of each assumption. This was done using the empirical data of real-world network flows. While it would certainly be interesting to extend this approach by using more diverse net-

work flow datasets, there are also other interesting extensions: Using synthetic flow data where the violation of each assumption can be controlled instead of using empirical flow data, would allow a more systematic evaluation of each assumption. However, an analytical approach instead of a data-driven simulation approach can also be considered. Furthermore, in this work, only two centrality measures were considered. A systematic extension to other centrality measures as well as to other types of network measures seems worthwhile. For example, there exist several clustering algorithms, such as the Girvan-Newman algorithm [GN02], based on shortest paths. Although further considerations will be required, it might be interesting to check whether incorporating real-world network flows into this type of methods yields intuitive results. For this approach, though, datasets containing a network flow and a ground truth about the cluster membership of each node is needed.

- During the analysis phase for the work presented in Chapter 4, we noticed that there are groups of nodes with similar ranking behaviors with respect to the centrality variants; for example, a subset of nodes that is not central with respect to standard centrality, but is central with respect to flow-based centrality variants. Thus, for this type of nodes, it seems that they are not embedded well in the network structure, but are relevant for the network flow. In future work, it might be interesting to formalize this concept which is similar to the concept of structural equivalence and identify nodes with a similar role with respect to the network flow.
- While this was not the scope of this thesis, tools for visualizing network flows were needed during the work on this thesis. To the best of our knowledge, there is no tool available that is able to visualize a network and its flow based on data. Providing such a tool would be extremely helpful for the whole research community.

Supplementary material

A.1 Supplementary material for Chapter 3 (Properties of network flows in complex networks)

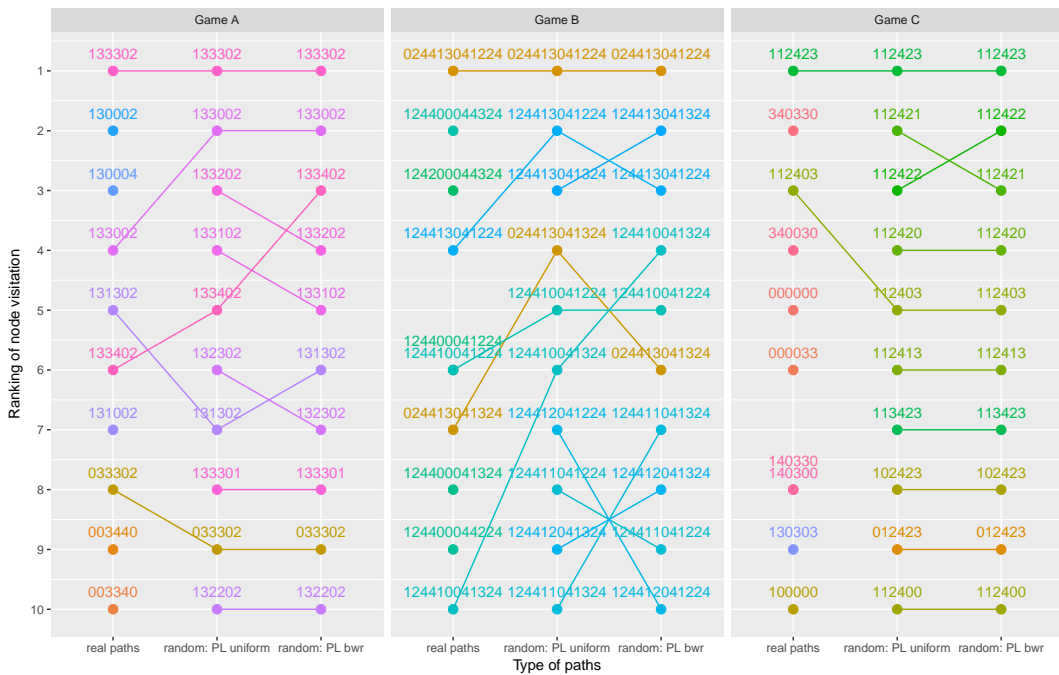
A.1.1 Nodes used most frequently by real trajectories and random walks

In Chapter 3, we compared the properties of several real-world network flows with the properties of random-walk-based models. Figure 3.11 on page 51 shows the node usage by the real-world trajectories and by the random-walk-based model variants as a scatter plot. As supplementary material, we provide Figure A.1 in which the ten most frequently used nodes of each trajectory variant (real trajectories and random walks) are shown for each dataset. For node usage by random walks, the average node usage over all $N = 500$ simulation runs is shown. The points are labeled with the node labels. For Rush Hour, the labels encode the cars' position on the board; thus, only node labels with a Hamming distance of 1 can belong to adjacent nodes.

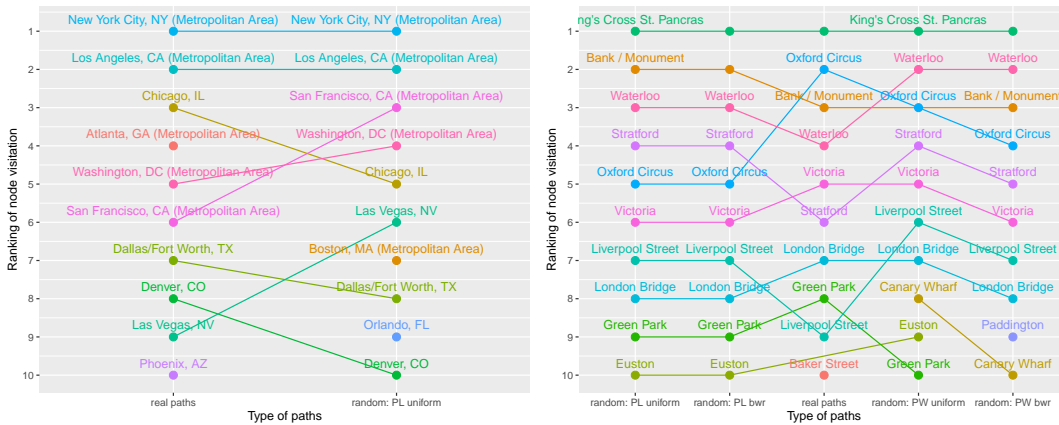
It can be seen that for the Rush Hour game instances, the nodes used most frequently by each trajectory variant do not show a large overlap: Besides the node representing the initial configuration, only direct neighbors of the initial configuration are found among the ten nodes used most frequently by *all* trajectory variants.

For the DB1B dataset, there are a lot more nodes that are among the ten nodes used most frequently by *both* trajectory types, real trajectories and random walks: Nine out of ten can be found in the ten nodes used most frequently by both trajectory types. A similar picture was found for the London Transport dataset (transitive graph): The node representing London's main station, King's Cross St. Pancras, shows the highest node usage by all types of trajectories. Apart from that, there are almost no nodes that occur in the ten nodes used most frequently for only one trajectory type.

For Wikispeedia, six out of the ten nodes used most frequently were found to be common for real trajectories and random walks. Interestingly, the node LATIN is the seventh most frequently used node by the random walks, although it is only the 73rd most frequently used node by real trajectories and is not adjacent to the most frequently used start nodes of the trajectories.

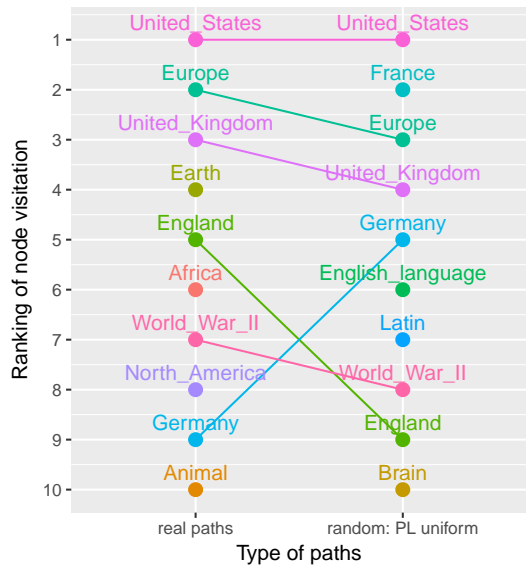


(a) Rush Hour



(b) DB1B

(c) London Transport



(d) Wikipedia

Figure A.1 Ten nodes used most frequently by real trajectories and random walks.

A.2 Supplementary material for Chapter 4 (Flow-based centrality measures)

A.2.1 Overlap of rankings

In Chapter 4, we introduced the measure weighted overlap of rankings (see Section 4.3 starting on page 70) for comparing two node rankings. In this measure, the number of common elements in the top x nodes of both rankings (their *overlap*) is considered for each possible x . This can be visualized by plotting the overlap of the two rankings as a function of x . Since the overlap function of two identical rankings is the identity function, the size of the area between the overlap function and the identity line then indicates the extent to which the rankings differ from each other. Furthermore, it can be seen in which part of the rankings, they exhibit their differences—rather among the lower-ranked nodes or also among the higher-ranked nodes. Figure A.2 shows this visualization for comparing the rankings of the flow-based betweenness measures introduced in Chapter 4 (see Section 4.2.1) to the corresponding ranking of standard betweenness centrality (including endpoints). The area between the overlap function and the identity line is colored blue, the maximal¹ area (realized by rankings where one is the reverse of the other) is colored gray.

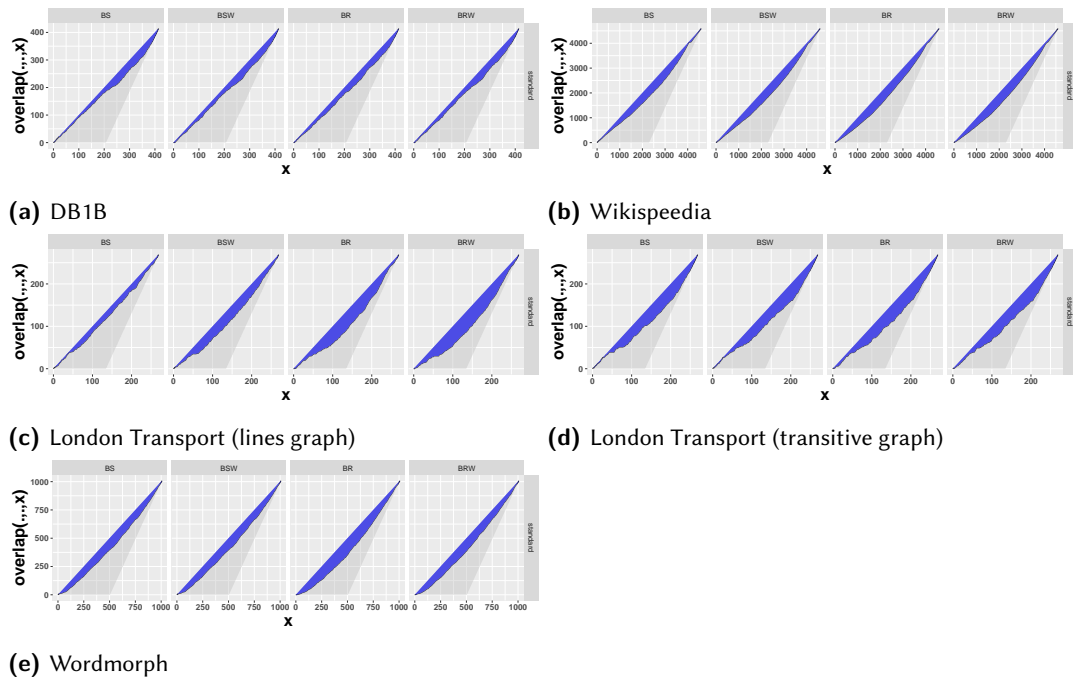


Figure A.2 Betweenness variants For each dataset and each flow-based betweenness variant, its overlap with the standard betweenness centrality is shown.

A.2.2 Variation of rankings of flow-based centralities

Figures A.3 (betweenness) and A.4 (closeness) visualize the deviation between the rankings of standard centrality (betweenness or closeness) and the flow-based centrality variants. For this purpose, the nodes are drawn on the x -axis (ordered by their ranking position with respect to standard centrality), and their ranking position is drawn on the y -axis separately for each centrality variant. Thus, the panels of the standard centrality measure show a straight line. For the flow-based centrality measures, it can be seen whether the nodes increase or decrease their ranking position

¹no ties assumed

compared to the standard centrality variant. Furthermore, for each node, the span of its ranking position is shown by a gray vertical line: For each node, the corresponding gray line ranges from the node's minimal to its maximal ranking position over all centrality variants.

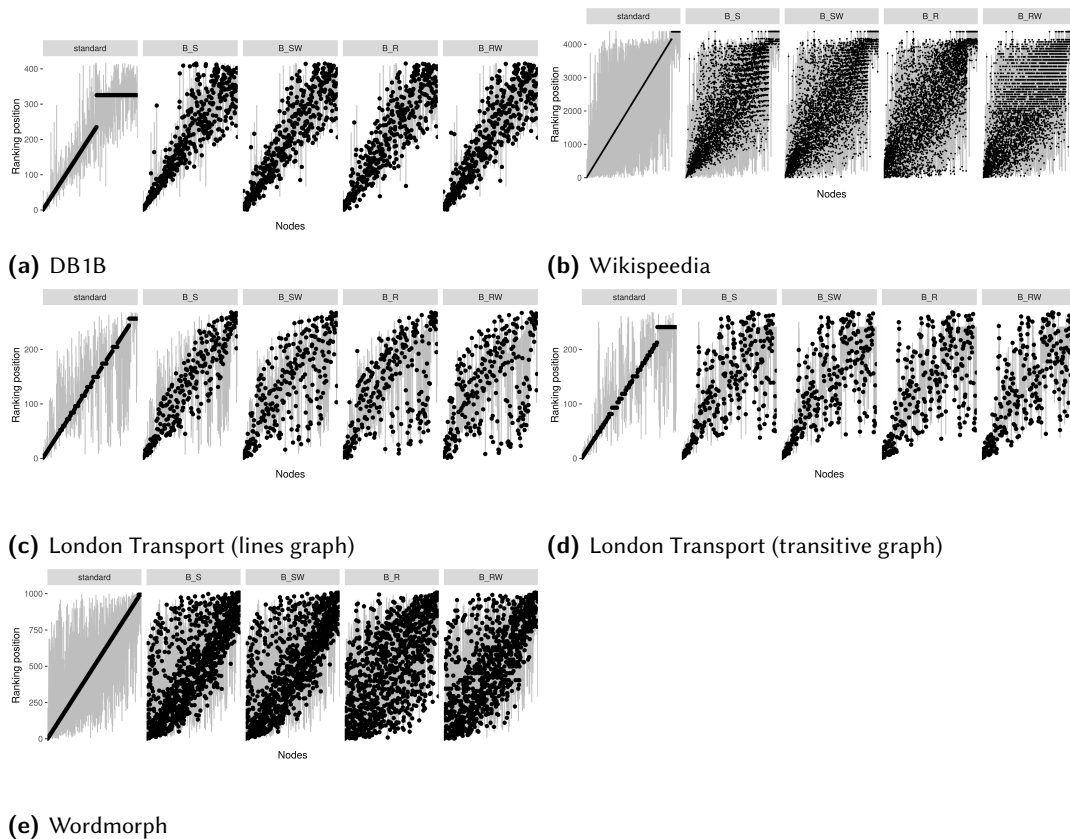


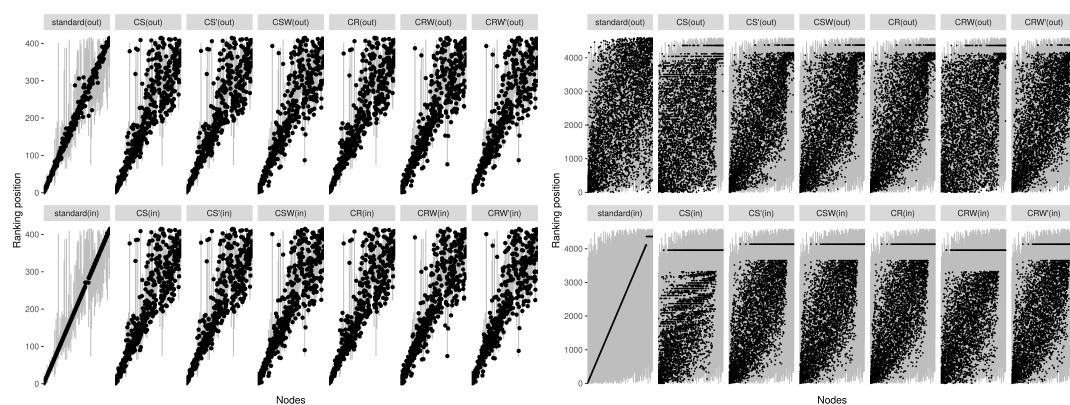
Figure A.3 Betweenness measures Ranking positions of each node with respect to each flow-based betweenness variant and standard betweenness centrality (including endpoints). The x -axis contains all nodes, ordered by their ranking position with respect to standard betweenness centrality (this order is the same for all panels of one subfigure); the y -axis represents the nodes' ranking position with respect to the corresponding centrality variant. For each node, a gray bar indicates the span of its ranking positions: A long gray bar indicates that a node changes ranking positions considerably from one variant to another.

A.2.3 Top ten nodes of flow-based centrality variants

In Chapter 4, we considered the ranking behavior of those nodes that were among the ten highest-ranked nodes with respect to any considered centrality variant. Figure 4.14 on page 92 shows this for all variants of the *out*-closeness measure. As supplementary material, Figure A.5 shows the behavior of the high-ranked nodes with respect to any *in*-closeness variant.

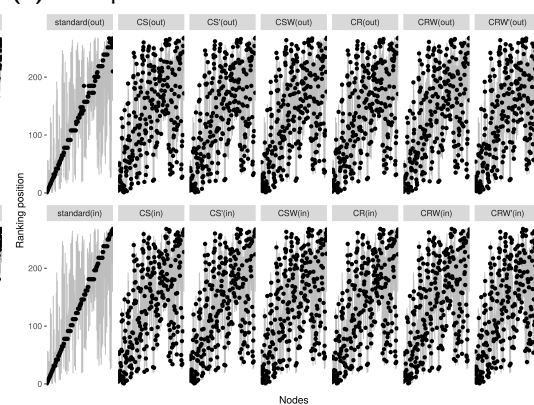
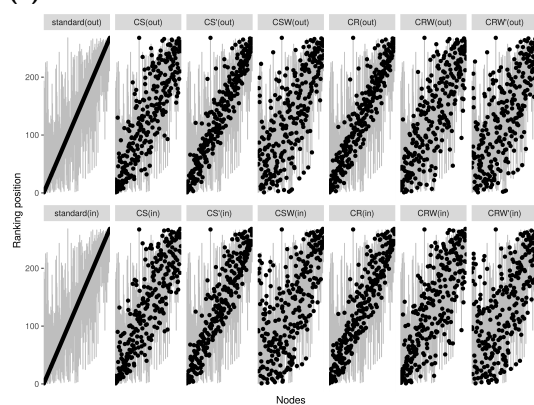
A.2.4 Min-max-ranking plot for flow-based closeness centralities

In Chapter 4, Figure 4.18 on page 97 shows for each node its minimal ranking position over all flow-based betweenness measures (and standard betweenness centrality) against its maximal ranking position over all measure variants. We additionally provide the corresponding figure for the flow-based closeness variants here. Nodes with large differences between their maximal and minimal ranking positions are shown with their node label.



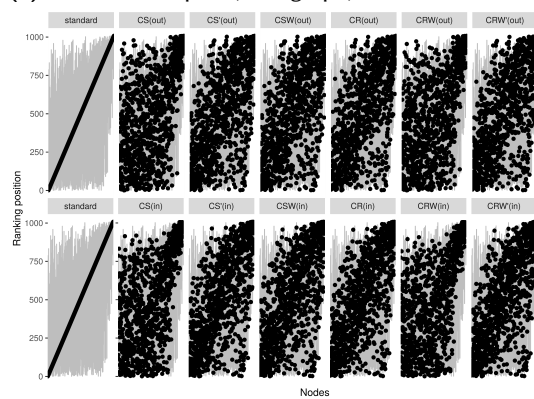
(a) DB1B

(b) Wikipedia



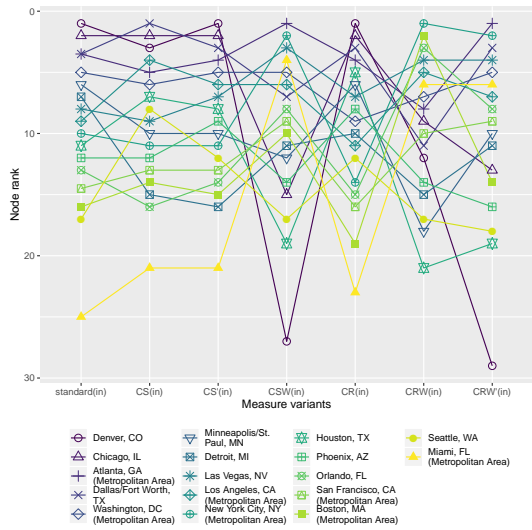
(c) London Transport (lines graph)

(d) London Transport (transitive graph)

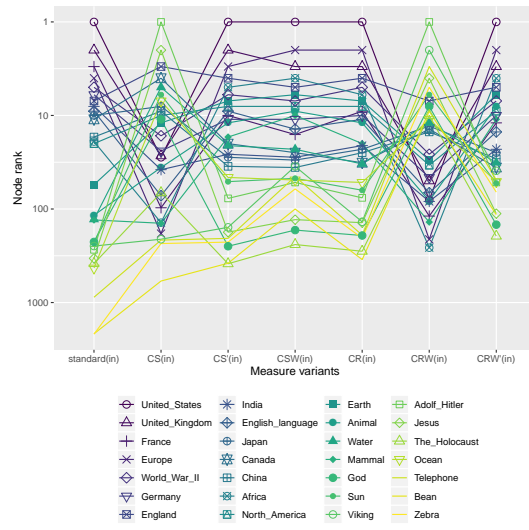


(e) Wordmorph

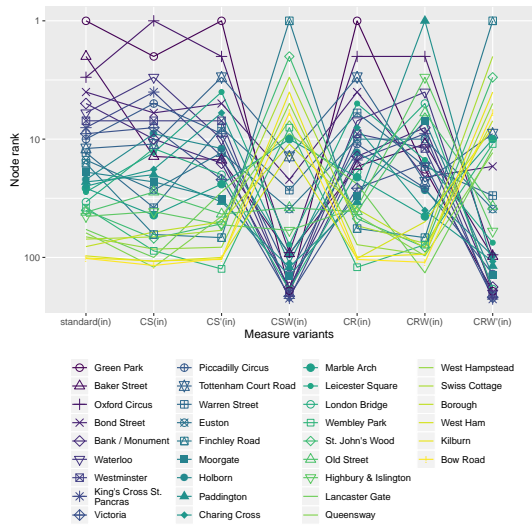
Figure A.4 Closeness measures Ranking positions of the nodes for standard in- and out-closeness and for all flow-based closeness measures. Each node's position on the x -axis is determined by its ranking position with respect to standard in-closeness. For each node, its minimal and maximal ranking positions with respect to any flow-based variant are depicted by a gray line.



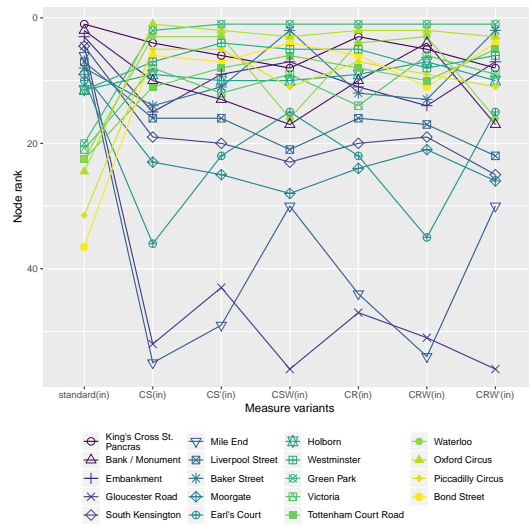
(a) DB1B.



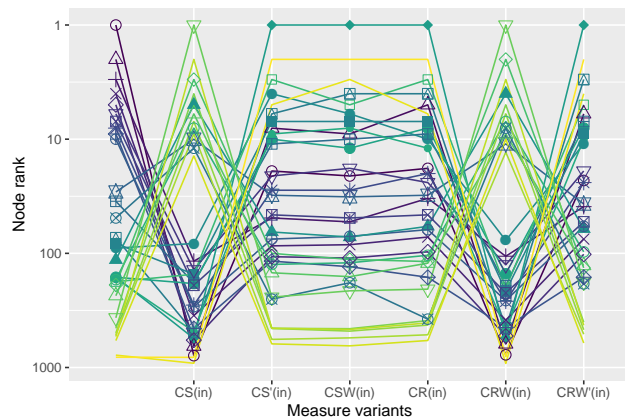
(b) Wikipedia. Note logarithmic scale on y -axis.



(c) London Transport (lines). Note logarithmic scale on y -axis.

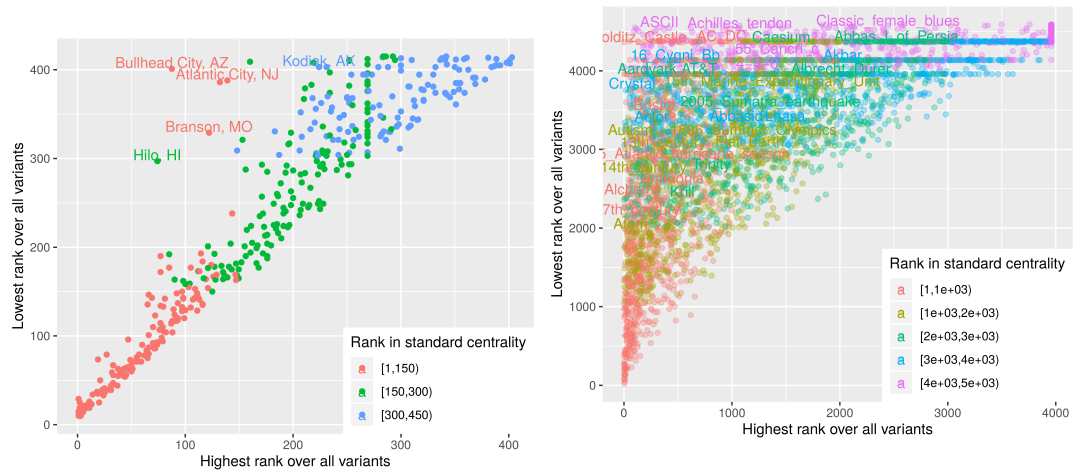


(d) London Transport (transitive). Note logarithmic scale on y -axis.



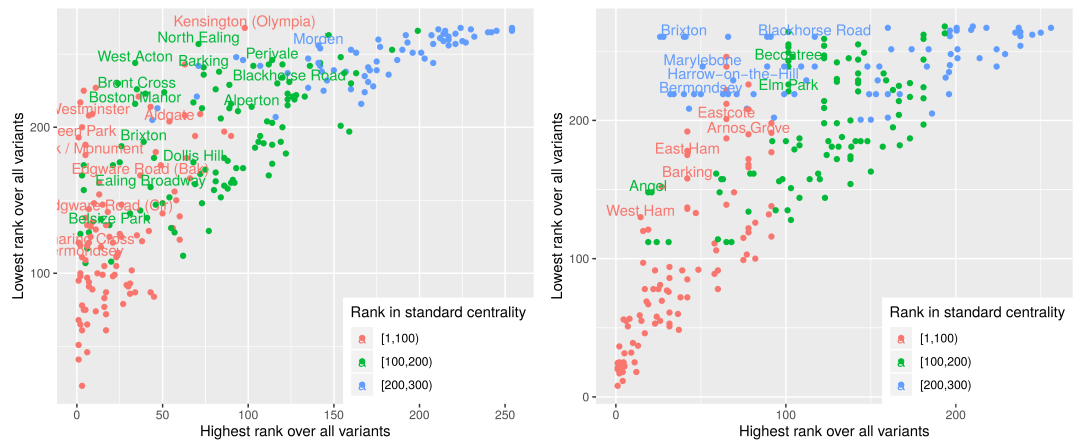
(e) Wordmorph. Note logarithmic scale on y -axis.

Figure A.5 Top 10 nodes (closeness) Ranking positions with respect to all (in)-closeness centrality variants of those nodes that are among the ten most central nodes with respect to at least one centrality variant. Top nodes have rank 1 (top of each plot). The order of the line colors is due to the nodes' standard closeness centrality.



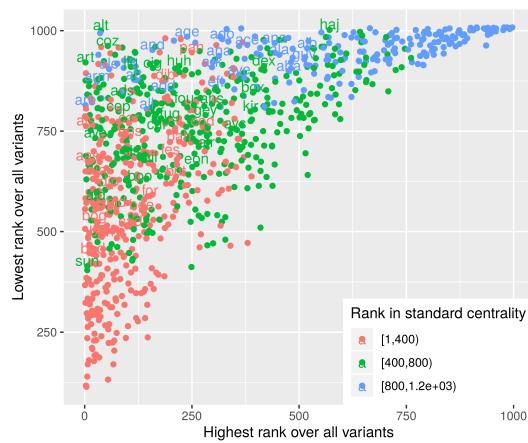
(a) DB1B

(b) Wikipedia



(c) London Transport (line graph)

(d) London Transport (transitive graph)



(e) Wordmorph

Figure A.6 Min-max-plot for rankings for flow-based closeness centrality variants: Each node is represented by a point, and its highest ranking over all closeness variants (flow-based and standard closeness centrality) against its lowest ranking position over all variants is shown. The color of the dots indicates the nodes' ranking with respect to standard out-closeness centrality.

A.3 Supplementary material for Chapter 5 (Summarizing trajectories of real-world network flows)

A.3.1 Properties of similarity and distance measures

In Chapter 5, several similarity and distance measures for walks in graphs were introduced. For each measure, Table A.1 shows its properties (without proof).

Table A.1 Properties of the similarity and distance measures for walks in graphs, introduced and used in Chapter 5.

		Boundedness	Non-negativity	Coincidence	Symmetry	Triangle inequality
Set	s_{nss}	✓	✓	✗	✓	✓
	s_{nss}^N	✓	✓	✗	✓	✓
	s_{ess}	✓	✓	✗	✓	✓
	s_{ess}^N	✓	✓	✗	✓	✓
Sequence	s_{lcs}	✗	✓	✗	✓	✓
	s_{lcs}^N	✓	✓	✗	✓	✓
Set of points in metric space	δ_h	✓	✓	✗	✓	✓
	δ_{mad}	✓	✓	✗	✓	✗
	δ_{mad}^N	✓	✓	✗	✓	✗
Sequence of points in metric space	δ_{dF}	✓	✓	✓	✓	✓
	δ_{adF}	✗	✓	✓	✓	✗
	δ_{adF}^N	✓	✓	✓	✓	✗
	δ_{sadF}	✗	✓	✓	✓	✗
	δ_{sadF}^N	✓	✓	✓	✓	✗

A.3.2 Algorithms for computing δ_{adF} and δ_{sadF}

In Chapter 5, variants of the discrete Fréchet distance were introduced for measuring the distance between two walks in a graph. These include the discrete Fréchet distance δ_{dF} , the additive discrete Fréchet distance δ_{adF} , and the simplified additive discrete Fréchet distance (see Sections 5.2.5 and 5.2.5). Eiter and Mannila provide an algorithm for computing δ_{dF} and (with a slight adaptation) δ_{adF} for polygonal curves P and Q , with a runtime of $\mathcal{O}(kl)$ where k and l denote the length of P and Q , based on a dynamic programming approach [EM94]. Algorithm A.1 shows the algorithm for computing δ_{adF} , given by Eiter and Mannila [EM94] where the notation was adapted to the notation used in this thesis.

Computing δ_{adF} The idea of the algorithm is as follows: A cost-optimal (left-total, right-total, and order-preserving) mapping between P and Q needs to be composed of cost-optimal mappings for sub-paths of P and Q . It is, however, not clear at which indices P and Q can be split into sub-paths such that the optimal mapping for sub-paths is included in the optimal mapping of P and Q . Therefore, in the dynamic programming approach, a table T is filled in which the costs of the mappings of the sub-paths at each possible “split index” are stored.

It is easy to verify that a cost-optimal mapping for the walks P and Q cannot contain any N-like structures: For a cost-optimal mapping containing the entries (i, j) , $(i + 1, j)$ and $(i + 1, j + 1)$ for some i, j , it is clear that the entry $(i + 1, j)$ can be removed from the mapping which still yields a left-total, right-total, and order-preserving mapping with smaller costs than the previous one. Thus, the assumed mapping containing such an N-like structure cannot be cost-optimal.

Algorithm A.1: Computing the additive discrete Fréchet distance δ_{adF} for two walks P and Q in a graph G , algorithm given by Eiter and Manilla [EM94] (slightly adapted, notation was adopted to the notation used in this thesis).

```

Data:  $G = (V, E)$ ,  $P = (p_1, p_2, \dots, p_l)$  and  $Q = (q_1, q_2, \dots, q_k)$ 
Result:  $cost(\mu)$  with  $\mu \in \mathcal{G}_{P,Q}$  optimal
//  $T(i, j)$  contains the costs of the optimal mapping for  $(p_1, \dots, p_i)$ 
// and  $(q_1, \dots, q_j)$ 
1  $T(1, 1) := d(p_1, q_1)$ ;
2 for  $i = 2$  to  $l$  do
3    $T(i, 1) := T(i - 1, 1) + d(p_i, q_1)$ ;
4 end
5 for  $j = 2$  to  $k$  do
6    $T(1, j) := T(1, j - 1) + d(p_1, q_j)$ ;
7   for  $i = 2$  to  $l$  do
8      $T(i, j) := d(p_i, q_j) + \min \{T(i - 1, j), T(i, j - 1), T(i - 1, j - 1)\}$ 
9   end
10 end
11 return  $T(l, k)$ ;

```

Based on the observation that an optimal mapping cannot contain N-like structures, it is clear that for any walks P and Q with $|P| > 1$ and $|Q| > 1^2$, the optimal mapping is composed of at least two node-disjoint parts: For a pair of indices (i, j) (also called *pair of separating indices*), all nodes to the left of i (and i itself) are mapped onto nodes to the left of j (or on j itself) and all nodes right of i are mapped onto node right of j . Thus, if these pairs of separating indices were known, the computation of the optimal mapping would be easy: Compute the optimal mappings for the sub-paths between the separating indices and merge them—which yields an optimal mapping for P and Q . Thus, in Algorithm A.1, a table T of $l \times k$ cells is filled such that the entry $T(i, j)$ contains the cost of the optimal mapping for the sub-paths $P_{1,i}$ and $Q_{1,j}$.

Computing δ_{sadF} In Chapter 5, a simplified version of δ_{adF} , i.e., δ_{sadF} , was introduced, in which the set of possible mappings is restricted to the set of left-total, right-total, order-preserving and right-unique mappings. Algorithm A.2 computes the simplified additive discrete Fréchet distance for two walks P and Q with $|P| = l \geq |Q| = k$ in $\mathcal{O}(k(l - k))$ and a space complexity of $\mathcal{O}(l - k)$ plus the costs of storing the graph G , under the assumption that the graph distances between the nodes are precomputed.

In the following, we assume that P and Q are walks with $l = |P| \geq |Q| = k$. Furthermore, for a given mapping μ , let $\mu_{i,\Delta} \subseteq \mu$ denote a submapping of μ that only contains those entries (i', j) with $i \leq i' \leq i + \Delta$.

We first note that several observations leading to the algorithm for δ_{adF} also hold for the possible mappings for δ_{sadF} : For δ_{sadF} , too, the optimal mapping cannot yield any N-like structure because a right-unique mapping cannot contain both (i, j) and (i, j') for any i, j, j' . Therefore, the optimal mapping of \mathcal{F} is also composed of optimal submappings along its pairs of separating indices. Hence, a dynamic programming approach such as the one for δ_{adF} can also be used to compute δ_{sadF} . However, since the mapping μ needs to be right-unique, it is no longer allowed that any $i \in I(P)$ is contained in more than one mapping link. Hence, for the computation of the costs of the optimal mapping for $P_{1,i}$ and $Q_{1,j}$, the link (i, j) is added either to the optimal mapping of $P_{1,i-1}$ and $Q_{1,j-1}$ or to the optimal mapping of $P_{1,i-1}$ and $Q_{1,j}$.

²Note that if $|P| = 1$ or $|Q| = 1$, there is only one possible left-total, right-total and order-preserving mapping.

Algorithm A.2: Computing the simplified additive discrete Fréchet distance for two walks P and Q in a graph G

Data: $G = (V, E)$, $P = (p_1, p_2, \dots, p_l)$ and $Q = (q_1, q_2, \dots, q_k)$ with $l \geq k$
Result: $\text{cost}(g)$ with $g \in \mathcal{F}_{P,Q}$ optimal
// $T(\Delta, j)$ contains the costs of the optimal mapping for $P_{1,j+\Delta}$ and $Q_{1,j}$

```

1  $T(0, 1) := d(p_1, q_1);$ 
2 for  $\Delta = 1$  to  $l - k$  do
3   |  $T(\Delta, 1) := T(\Delta - 1, 1) + d(p_{1+\Delta}, q_1);$ 
4 end
5 for  $j = 2$  to  $k$  do
6   |  $T(0, j) = d(p_j, q_j) + T(0, j - 1);$ 
7   | for  $\Delta = 1$  to  $l - k$  do
8     |  $T(\Delta, j) := d(p_{j+\Delta}, q_j) + \min \{T(\Delta - 1, j), T(\Delta, j - 1)\}$ 
9     | end
10 end
11 return  $T(l - k, k);$ 

```

However, for these mappings, the number of possible pairs of separating indices can be reduced: Since the mapping needs to be right-unique and left-total and right-total, for each $j \in I(Q)$, there is only a limited range of indices in $I(Q)$ with which j can form a pair of separating indices in an optimal mapping. Informally, each $i \in I(P)$ can only be used once in the mapping, but all $j \in I(Q)$ need to be "hit" by an i at least once. If a $j \in I(Q)$ is hit by too many $i \in I(P)$, there are not enough $i \in I(P)$ left to cover the remaining indices of Q .

Lemma 1

If (i, j) is a pair of separating indices for a mapping $\mu \in \mathcal{F}$ and $j \in \{1, \dots, k - 1\}$, then $j \leq i \leq j + l - k$. If (i, j) is a pair of separating indices and $j = k$, then $i = l$.

Proof. Let $\mu \in \mathcal{F}$ be a mapping for P and Q and (i, j) a pair of separating indices. If $j = k$, then it is clear that $i = l$, otherwise μ cannot be left-total and order-preserving. Let $j < k$. We first prove the first part of the inequality, i.e., $j \leq i$. For the sake of a contradiction, assume $i < j$. The fact that (i, j) is a pair of separating indices for μ implies that $\mu_{1,i}$ is a left-total, right-total, right-unique and order-preserving mapping for $P_{1,i}$ and $Q_{1,j}$. Hence, $\mu_{1,i}$ maps i nodes onto j nodes. If $i < j$ holds, $\mu_{1,i}$ cannot be right-total and right-unique. Thus, it needs to hold $j \leq i$. For the second part of the inequality, i.e., $i \leq j + l - k$, we can use the same argumentation. If (i, j) is a pair of separating indices, $\mu_{i+1,l}$ is a right-total, left-total, right-unique, and order-preserving mapping for $P_{i+1,l}$ and $Q_{j+1,k}$. Assume $i > j + l - k$, then $\mu_{i+1,l}$ maps $l - i < l - (j + l - k) = k - j$ nodes onto $k - j$ nodes, which is not possible in a right-total and right-unique way. \square

The lemma implies that not all (i, j) are candidates for being a pair of separating indices, thus, in the algorithm computing δ_{sadF} , not all (i, j) need to be tested, but only those (i, j) that fulfill Lemma 1. This leads to Algorithm A.2. Note that for the computation of an entry in column j , only one entry in column $j - 1$ and one entry in column j which were computed before are needed. Therefore, it is sufficient to keep one column of T in the memory instead of the complete table T . This reduces the memory requirement to a matrix of size $(l - k + 1) \times 1$ (plus the memory requirement for storing the graph).

Bibliography

- [AA05] Lada Adamic and Eytan Adar. How to search a social network. *Social Networks*, 27(3):187–203, 2005. doi:[10.1016/j.socnet.2005.01.007](https://doi.org/10.1016/j.socnet.2005.01.007).
- [AAKS14] Pankaj K Agarwal, Rinat Ben Avraham, Haim Kaplan, and Micha Sharir. Computing the Discrete Fréchet Distance in Subquadratic Time. *SIAM Journal on Computing*, 43(2):429–449, 2014. doi:<https://doi.org/10.1137/130920526>.
- [Adm17] Federal Aviation Administration. *National Plan of Integrated Airport Systems (NPIAS) 2017-2012*. U.S. Department of Transportation, Report of the Secretary of Transportation to the United States Congress edition, 2017. URL http://www.faa.gov/airports/planning_capacity/npias/reports/.
- [ADS12] Serge Autexier, Dominik Dietrich, and Marvin Schiller. Towards an intelligent tutor for mathematical proofs. *Electronic Proceedings in Theoretical Computer Science*, 79:1–28, Feb 2012. doi:[10.4204/eptcs.79.1](https://doi.org/10.4204/eptcs.79.1).
- [AG95] Helmut Alt and Michael Godau. Computing the Fréchet Distance between Two Polygonal Curves. *International Journal of Computational geometry & Applications*, 5:75–91, 1995. doi:[10.1142/S0218195995000064](https://doi.org/10.1142/S0218195995000064).
- [AGLW07] Mattias Andersson, Joachim Gudmundsson, Patrick Laube, and Thomas Wolle. Reporting leaders and followers among trajectories of moving point objects. *Geoinformatica*, 12(4):497–528, oct 2007. doi:[10.1007/s10707-007-0037-9](https://doi.org/10.1007/s10707-007-0037-9).
- [ALF12] Lada A Adamic, Thomas M Lento, and Andrew T Fiore. How you met me. In *Sixth International AAAI Conference on Weblogs and Social Media*, 2012.
- [And93] J. R. Anderson. Problem Solving and Learning. *American Psychologist*, 48, 1993.
- [Ant71] Jac M Anthonisse. The rush in a directed graph. *Stichting Mathematisch Centrum. Mathematische Besliskunde*, BN 9/71:1–10, 1971.
- [AT06] G. Antonini and J.P. Thiran. Counting Pedestrians in Video Sequences Using Trajectory Clustering. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(8):1008–1020, aug 2006. doi:[10.1109/tcsvt.2006.879118](https://doi.org/10.1109/tcsvt.2006.879118).
- [BA99] Albert-László Barabási and Réka Albert. Emergence of Scaling in Random Networks. *Science*, 286(5439):509–512, 1999. doi:[10.1126/science.286.5439.509](https://doi.org/10.1126/science.286.5439.509).
- [Bav48] Alex Bavelas. A mathematical model for group structures. *Human Organization*, 7(3):16–30, 1948.

- [Bav50] Alex Bavelas. Communication patterns in task-oriented groups. *The journal of the acoustical society of America*, 22(6):725–730, 1950.
- [BBG⁺11] Kevin Buchin, Maïke Buchin, Joachim Gudmundsson, Maarten Löffler, and Jun Luo. Detecting commuting patterns by clustering subtrajectories. *International Journal of Computational Geometry & Applications*, 21(03):253–282, 2011. doi:[10.1142/S0218195911003652](https://doi.org/10.1142/S0218195911003652).
- [BBR⁺12] Lars Backstrom, Paolo Boldi, Marco Rosa, Johan Ugander, and Sebastiano Vigna. Four degrees of separation. In *Proceedings of the 3rd Annual ACM Web Science Conference on - WebSci '12*. ACM Press, 2012. doi:[10.1145/2380718.2380723](https://doi.org/10.1145/2380718.2380723).
- [BBRK14] Stefano Bennati, Sven Brussow, Marco Ragni, and Lars Konieczny. Gestalt effects in planning: Rush-hour as an example. In *Proceedings of the Cognitive Science Society*, volume 36, 2014.
- [BC19] Anna D. Broido and Aaron Clauset. Scale-free networks are rare. *Nature Communications*, 10(1), mar 2019. doi:[10.1038/s41467-019-08746-5](https://doi.org/10.1038/s41467-019-08746-5).
- [BE05] Ulrik Brandes and Thomas Erlebach, editors. *Network Analysis*. Springer, Berlin Heidelberg, 2005. doi:[10.1007/b106453](https://doi.org/10.1007/b106453).
- [BE06] Stephen P. Borgatti and Martin G. Everett. A graph-theoretic perspective on centrality. *Social Networks*, 28(4):466 – 484, 2006. doi:[10.1016/j.socnet.2005.11.005](https://doi.org/10.1016/j.socnet.2005.11.005).
- [Bea65] Murray A Beauchamp. An improved index of centrality. *Behavioral science*, 10(2):161–163, 1965.
- [BEJ13] Stephen P Borgatti, Martin G Everett, and Jeffrey C Johnson. *Analyzing social networks*. Sage, 2013.
- [BFSO98] Leo Breiman, Jerome Friedman, Charles J Stone, and R A Olshen. *Classification and Regression Trees*. The Wadsworth and Brooks-Cole Statistics-Probability Series. Taylor & Francis, 198.
- [BGL10] Albert-László Barabási, Natali Gulbahce, and Joseph Loscalzo. Network medicine: a network-based approach to human disease. *Nature Reviews Genetics*, 12(1):56–68, dec 2010. doi:[10.1038/nrg2918](https://doi.org/10.1038/nrg2918).
- [BKS03] Faisal Bashir, Ashfaq Khokhar, and Dan Schonfeld. Segmented trajectory based indexing and retrieval of video data. In *Proceedings of the International Conference on Image Processing*, volume 2, pages II–623. IEEE, 2003. doi:[10.1109/ICIP.2003.1246757](https://doi.org/10.1109/ICIP.2003.1246757).
- [BL11] Lars Backstrom and Jure Leskovec. Supervised random walks. In *Proceedings of the fourth ACM international conference on Web search and data mining - WSDM '11*. ACM Press, 2011. doi:[10.1145/1935826.1935914](https://doi.org/10.1145/1935826.1935914).
- [BLM⁺06] S Boccaletti, V Latora, Y Moreno, M Chavez, and D-U Hwang. Complex networks: Structure and dynamics. *Physics Reports*, 424(4-5):175–308, 2006. doi:[10.1016/j.physrep.2005.10.009](https://doi.org/10.1016/j.physrep.2005.10.009).
- [BMBL09] S. P. Borgatti, A. Mehra, D. J. Brass, and G. Labianca. Network analysis in the social sciences. *Science*, 323(5916):892–895, feb 2009. doi:[10.1126/science.1165821](https://doi.org/10.1126/science.1165821).

- [BO04] Albert-László Barabási and Zoltán N. Oltvai. Network biology: understanding the cell's functional organization. *Nature Reviews Genetics*, 5(2):101–113, feb 2004. doi:[10.1038/nrg1272](https://doi.org/10.1038/nrg1272).
- [Boc15] Mareike Bockholt. Measures for the similarity of paths in complex networks. Master's thesis, TU Kaiserslautern, 2015.
- [Bol01] Béla Bollobás. *Random Graphs*. Cambridge University Press, aug 2001. doi:[10.1017/cbo9780511814068](https://doi.org/10.1017/cbo9780511814068).
- [Bon72] Phillip Bonacich. Factoring and weighting approaches to status scores and clique identification. *Journal of Mathematical Sociology*, 2(1):113–120, 1972.
- [Bon87] Phillip Bonacich. Power and centrality: A family of measures. *American journal of sociology*, 92(5):1170–1182, 1987.
- [Bor05] Stephen P. Borgatti. Centrality and network flow. *Social Networks*, 27(1):55–71, 2005. doi:<http://dx.doi.org/10.1016/j.socnet.2004.11.008>.
- [Bra46] Auguste Bravais. Sur les probabilités des erreurs de situation d'un point. *Mémoires de l'Académie Royale des Sciences de l'Institut de France*, 9:255–332, 1846.
- [BRMA12] Eytan Bakshy, Itamar Rosenn, Cameron Marlow, and Lada Adamic. The role of social networks in information diffusion. In *Proceedings of the 21st International Conference on World Wide Web, WWW '12*, pages 519–528, New York, NY, USA, 2012. ACM. doi:[10.1145/2187836.2187907](https://doi.org/10.1145/2187836.2187907).
- [BRMW13] Ulrik Brandes, Garry Robins, Ann McCranie, and Stanley Wasserman. What is network science? *Network Science*, 1(1):1–15, 2013. doi:[10.1017/nws.2013.2](https://doi.org/10.1017/nws.2013.2).
- [BSK04] Dan Buzan, Stan Sclaroff, and George Kollios. Extraction and clustering of motion trajectories in video. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 2, pages 521–524. IEEE, 2004. doi:[10.1109/ICPR.2004.1334287](https://doi.org/10.1109/ICPR.2004.1334287).
- [Bur16] Bureau of Transportation Statistics. Airline Origin and Destination Survey (DB1B). <https://www.transtats.bts.gov/DataIndex.asp>, 2016.
- [But09] Carter T. Butts. Revisiting the foundations of network analysis. *Science*, 325(5939):414–416, jul 2009. doi:[10.1126/science.1171022](https://doi.org/10.1126/science.1171022).
- [BZ15] Mareike Bockholt and Katharina Anna Zweig. Why Is This So Hard? Insights from the State Space of a Simple Board Game. In Stefan Göbel, Minhua Ma, Jannicke Baalsrud Hauge, Manuel Fradinho Oliveira, Josef Wiemeyer, and Viktor Wendel, editors, *Serious Games*, pages 147–157, Cham, 2015. Springer International Publishing. doi:[10.1007/978-3-319-19126-3_13](https://doi.org/10.1007/978-3-319-19126-3_13).
- [BZ16] Mareike Bockholt and Katharina A. Zweig. Clustering of paths in complex networks. In *Studies in Computational Intelligence*, pages 183–195. Springer International Publishing, nov 2016. doi:[10.1007/978-3-319-50901-3_15](https://doi.org/10.1007/978-3-319-50901-3_15).
- [Cen10] D. Centola. The spread of behavior in an online social network experiment. *Science*, 329(5996):1194–1197, sep 2010. doi:[10.1126/science.1185231](https://doi.org/10.1126/science.1185231).
- [CF07] Nicholas A. Christakis and James H. Fowler. The spread of obesity in a large social

- network over 32 years. *New England Journal of Medicine*, 357(4):370–379, jul 2007. doi:[10.1056/nejmsa066082](https://doi.org/10.1056/nejmsa066082).
- [CF08] Nicholas A. Christakis and James H. Fowler. The collective dynamics of smoking in a large social network. *New England Journal of Medicine*, 358(21):2249–2258, 2008, <https://doi.org/10.1056/NEJMsa0706154>. doi:[10.1056/NEJMsa0706154](https://doi.org/10.1056/NEJMsa0706154).
- [CKR⁺17] Attila Csoma, Attila Kőrösi, Gábor Rétvári, Zalán Heszberger, József Bíró, Mariann Slíz, Andrea Avena-Koenigsberger, Alessandra Griffa, Patric Hagmann, and András Gulyás. Routes obey hierarchy in complex networks. *Scientific Reports*, 7(1), 2017. doi:[10.1038/s41598-017-07412-4](https://doi.org/10.1038/s41598-017-07412-4).
- [CKRS12] Flavio Chierichetti, Ravi Kumar, Prabhakar Raghavan, and Tamas Sarlos. Are web users really markovian? In *Proceedings of the 21st International Conference on World Wide Web, WWW '12*, page 609–618, New York, NY, USA, 2012. Association for Computing Machinery. doi:[10.1145/2187836.2187919](https://doi.org/10.1145/2187836.2187919).
- [CM10] Kwangsu Cho and Charles MacArthur. Student revision with peer and expert reviewing. *Learning and Instruction*, 20(4):328–338, aug 2010. doi:[10.1016/j.learninstruc.2009.08.006](https://doi.org/10.1016/j.learninstruc.2009.08.006).
- [CMVR87] Alfonso Caramazza, Gabriele Miceli, Giampiero Villa, and Cristina Romani. The role of the graphemic buffer in spelling: Evidence from a case of acquired dysgraphia. *Cognition*, 26(1):59–85, 1987. doi:[10.1016/0010-0277\(87\)90014-X](https://doi.org/10.1016/0010-0277(87)90014-X).
- [CV12] Lidia Ceriani and Paolo Verme. The origins of the Gini index: extracts from Variabilità e Mutabilità (1912) by Corrado Gini. *The Journal of Economic Inequality*, 10(3):421–443, 2012.
- [Dam64] Fred J. Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176, mar 1964. doi:[10.1145/363958.363994](https://doi.org/10.1145/363958.363994).
- [Dav8] Mark Davis. The Corpus of Contemporary American English (COCA): 600 million words, 1990-present. Available online at <https://www.english-corpora.org/coca/>, 2008-.
- [DCMHW10] Munmun De Choudhury, Winter A. Mason, Jake M. Hofman, and Duncan J. Watts. Inferring relevant social networks from interpersonal communication. In *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, page 301–310, New York, NY, USA, 2010. Association for Computing Machinery. doi:[10.1145/1772690.1772722](https://doi.org/10.1145/1772690.1772722).
- [DDLMM13] Manlio De Domenico, Antonio Lima, Paul Mougél, and Mirco Musolesi. The anatomy of a scientific rumor. *Scientific reports*, 3:2980, 2013. doi:[10.1038/srep02980](https://doi.org/10.1038/srep02980).
- [DLZ12] Isadora Dorn, Andreas Lindenblatt, and Katharina A. Zweig. The trilemma of network analysis. In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 9–14, Washington, DC, USA, 2012. doi:<https://doi.org/10.1109/ASONAM.2012.12>.
- [Dod03] P. S. Dodds. An experimental study of search in global social networks. *Science*, 301(5634):827–829, aug 2003. doi:[10.1126/science.1081058](https://doi.org/10.1126/science.1081058).

- [DPV05] Imre Derényi, Gergely Palla, and Tamás Vicsek. Clique percolation in random networks. *Physical Review Letters*, 94(16), apr 2005. doi:[10.1103/physrevlett.94.160202](https://doi.org/10.1103/physrevlett.94.160202).
- [EM94] Thomas Eiter and Heikki Mannila. Computing discrete Fréchet distance. Technical report, Information Systems Department, Technical University of Vienna, 1994.
- [EM97] Thomas Eiter and Heikki Mannila. Distance measures for point sets and their computation. *Acta Informatica*, 34(2):109–133, 1997.
- [ENSM12] A. Eichelmann, S. Narciss, L. Schnaubert, and E. Melis. Typische Fehler bei der Addition und Subtraktion von Brüchen—Ein Review zu empirischen Fehleranalysen. *Journal für Mathematik-Didaktik*, 33(1):29–57, 2012. doi:[10.1007/s13138-011-0031-5](https://doi.org/10.1007/s13138-011-0031-5).
- [ER60] Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci.*, 5(1):17–60, 1960.
- [Eul41] Leonhard Euler. Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, pages 128–140, 1741.
- [FBW91] Linton C Freeman, Stephen P Borgatti, and Douglas R White. Centrality in valued graphs: A measure of betweenness based on network flow. *Social networks*, 13(2):141–154, 1991. doi:[10.1016/0378-8733\(91\)90017-N](https://doi.org/10.1016/0378-8733(91)90017-N).
- [For10] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75 – 174, 2010. doi:<https://doi.org/10.1016/j.physrep.2009.11.002>.
- [Fré06] M. Maurice Fréchet. Sur quelques points du calcul fonctionnel. *Rendiconti del Circolo Matematico di Palermo (1884-1940)*, 22(1):1–72, 1906.
- [Fre77] Linton C Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.
- [Fre78] Linton C Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1978. doi:[10.1016/0378-8733\(78\)90021-7](https://doi.org/10.1016/0378-8733(78)90021-7).
- [Fri83] N. E. Friedkin. Horizons of observability and limits of informal control in organizations. *Social Forces*, 62(1):54–77, sep 1983. doi:[10.1093/sf/62.1.54](https://doi.org/10.1093/sf/62.1.54).
- [FRM79] Linton C Freeman, Douglas Roeder, and Robert R Mulholland. Centrality in social networks: II. Experimental results. *Social networks*, 2(2):119–141, 1979.
- [Gal89] Francis Galton. I. co-relations and their measurement, chiefly from anthropometric data. *Proceedings of the Royal Society of London*, 45(273-279):135–145, 1889.
- [GBR⁺20] András Gulyás, József Bíró, Gábor Rétvári, Márton Novák, Attila Kőrösi, Mariann Slíz, and Zalán Heszberger. The role of detours in individual human navigation patterns of complex networks. *Scientific Reports*, 10(1), jan 2020. doi:[10.1038/s41598-020-57856-4](https://doi.org/10.1038/s41598-020-57856-4).
- [GH17] Joachim Gudmundsson and Michael Horton. Spatio-temporal analysis of team sports. *ACM Computing Surveys*, 50(2):1–34, apr 2017. doi:[10.1145/3054132](https://doi.org/10.1145/3054132).
- [GHFZ13] Adrien Guille, Hakim Hacid, Cecile Favre, and Djamel A. Zighed. Information diffusion in online social networks: A survey. *SIGMOD Rec.*, 42(2):17–28, July 2013.

doi:[10.1145/2503792.2503797](https://doi.org/10.1145/2503792.2503797).

- [Gin12] Corrado Gini. *Variabilità e Mutuabilità*. Contributo allo Studio delle Distribuzioni e delle Relazioni Statistiche, Bologna, 1912.
- [Gin21] Corrado Gini. Measurement of inequality of incomes. *The Economic Journal*, 31(121):124–126, 1921. URL <http://www.jstor.org/stable/2223319>.
- [GMTA05] Roger Guimerà, Stefano Mossa, Adrian Turttschi, and LA Nunes Amaral. The world-wide air transportation network: Anomalous centrality, community structure, and cities' global roles. *Proceedings of the National Academy of Sciences*, 102(22):7794–7799, 2005. doi:[10.1073/pnas.0407994102](https://doi.org/10.1073/pnas.0407994102).
- [GN02] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, jun 2002. doi:[10.1073/pnas.122653799](https://doi.org/10.1073/pnas.122653799).
- [Gre78] James G Greeno. Natures of problem-solving abilities. *Handbook of learning and cognitive processes*, 5:239–270, 1978.
- [Gus97] Dan Gusfield. *Algorithms on strings, trees and sequences: computer science and computational biology*. Cambridge University Press, New York, NY, USA, 1997.
- [GW02] Lixin Gao and Feng Wang. The extent of AS path inflation by routing policies. In *Global Telecommunications Conference, 2002. GLOBECOM '02*. IEEE, 2002. doi:[10.1109/glocom.2002.1189018](https://doi.org/10.1109/glocom.2002.1189018).
- [Hau14] Felix Hausdorff. *Grundzüge der Mengenlehre*. Veit and Company, Leipzig, 1914. Available online at <https://archive.org/details/grundzgedermen00hausuoft>.
- [HS12] Petter Holme and Jari Saramäki. Temporal networks. *Physics Reports*, 519(3):97–125, oct 2012. doi:[10.1016/j.physrep.2012.03.001](https://doi.org/10.1016/j.physrep.2012.03.001).
- [HSK⁺13] Aniko Hannak, Piotr Sapiezynski, Arash Molavi Kakhki, Balachander Krishnamurthy, David Lazer, Alan Mislove, and Christo Wilson. Measuring personalization of web search. In *Proceedings of the 22nd international conference on World Wide Web - WWW '13*. ACM Press, 2013. doi:[10.1145/2488388.2488435](https://doi.org/10.1145/2488388.2488435).
- [IM09] José Luis Iribarren and Esteban Moro. Impact of human activity patterns on the dynamics of information diffusion. *Phys. Rev. Lett.*, 103:038702, Jul 2009. doi:[10.1103/PhysRevLett.103.038702](https://doi.org/10.1103/PhysRevLett.103.038702).
- [Jac12] Paul Jaccard. The distribution of the flora in the alpine zone. *New phytologist*, 11(2):37–50, 1912.
- [JCR64] AG Juilland and E Chang-Rodriguez. *Frequency dictionary of Spanish words*. Mouton de Gruyter, The Hague, 1964.
- [JJS04] Imran N Junejo, Omar Javed, and Mubarak Shah. Multi feature path modeling for video surveillance. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 2, pages 716–719. IEEE, 2004. doi:[10.1109/ICPR.2004.1334359](https://doi.org/10.1109/ICPR.2004.1334359).
- [JP12] Petr Jarušek and Radek Pelánek. Analysis of a Simple Model of Problem Solving Times. In Cerri, Stefano A. and Clancey, WilliamJ. and Papadourakis, Giorgos and

- Panourgia, Kitty, editor, *Intelligent Tutoring Systems*, volume 7315 of *Lecture Notes in Computer Science*, pages 379–388. Springer, Berlin Heidelberg, 2012. doi:[10.1007/978-3-642-30950-2_49](https://doi.org/10.1007/978-3-642-30950-2_49).
- [Kat53] Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.
- [KCR⁺18] Attila Kőrösi, Attila Csoma, Gábor Rétvári, Zalán Heszberger, József Bíró, János Tapolcai, István Pelle, Dávid Klajbár, Márton Novák, Valentina Halasi, and András Gulyás. A dataset on human navigation strategies in foreign networked systems. *Scientific Data*, 5(1), mar 2018. doi:[10.1038/sdata.2018.37](https://doi.org/10.1038/sdata.2018.37).
- [Ken38] M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, jun 1938. doi:[10.1093/biomet/30.1-2.81](https://doi.org/10.1093/biomet/30.1-2.81).
- [KLP⁺05] Dirk Koschützki, Katharina Anna Lehmann, Leon Peeters, Stefan Richter, Dagmar Tenfelde-Podehl, and Oliver Zlotowski. Centrality indices. In Ulrik Brandes and Thomas Erlebach, editors, *Network Analysis: Methodological Foundations*, volume 3418 of *Lecture Notes in Computer Science*, pages 16–61. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005. doi:[10.1007/978-3-540-31955-9_3](https://doi.org/10.1007/978-3-540-31955-9_3).
- [Kos19] Sven Kosub. A note on the triangle inequality for the jaccard distance. *Pattern Recognition Letters*, 120:36–38, Apr 2019. doi:[10.1016/j.patrec.2018.12.007](https://doi.org/10.1016/j.patrec.2018.12.007).
- [LBF05] Jeffrey S Larson, Eric T Bradlow, and Peter S Fader. An exploratory look at super-market shopping paths. *International Journal of research in Marketing*, 22(4):395–414, 2005. doi:[10.1016/j.ijresmar.2005.09.005](https://doi.org/10.1016/j.ijresmar.2005.09.005).
- [Lea94] Lucian L Leape. Error in medicine. *Journal of the American Medical Association*, 272(23):1851–1857, 1994. doi:[10.1001/jama.1994.03520230061039](https://doi.org/10.1001/jama.1994.03520230061039).
- [Lev66] Vladimir I. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet physics doklady*, 10(8):707–710, 1966.
- [LH12] Sang Hoon Lee and Petter Holme. Exploring maps with greedy navigators. *Physical Review Letters*, 108(12), mar 2012. doi:[10.1103/physrevlett.108.128701](https://doi.org/10.1103/physrevlett.108.128701).
- [LHW07] Jae-Gil Lee, Jiawei Han, and Kyu-Young Whang. Trajectory clustering: a partition-and-group framework. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 593–604. ACM, 2007. doi:[10.1145/1247480.1247546](https://doi.org/10.1145/1247480.1247546).
- [LK14] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. <http://snap.stanford.edu/data>, June 2014.
- [LMP83] Edward O Laumann, Peter V Marsden, and David Prensky. The boundary specification problem in network analysis. In Linton C Freeman, Douglas R White, and A Kimball Romney, editors, *Research methods in social network analysis*. Transaction Publishers, New Brunswick, New Jersey, 1983.
- [LNK07] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031, 2007. doi:[10.1002/asi.20591](https://doi.org/10.1002/asi.20591).

- [LW71] François Lorrain and Harrison C. White. Structural equivalence of individuals in social networks. *The Journal of Mathematical Sociology*, 1(1):49–80, jan 1971. doi:[10.1080/0022250x.1971.9989788](https://doi.org/10.1080/0022250x.1971.9989788).
- [MAC15] E.J. Manley, J.D. Addison, and T. Cheng. Shortest path or anchor-based route choice: a large-scale empirical analysis of minicab routing in london. *Journal of Transport Geography*, 43:123 – 139, 2015. doi:<https://doi.org/10.1016/j.jtrangeo.2015.01.006>.
- [Man15] Ed Manley. Estimating urban traffic patterns through probabilistic interconnectivity of road network junctions. *PLOS ONE*, 10(5):e0127095, may 2015. doi:[10.1371/journal.pone.0127095](https://doi.org/10.1371/journal.pone.0127095).
- [ME02] Dimitrios Makris and Tim Ellis. Path detection in video surveillance. *Image and Vision Computing*, 20(12):895–903, 2002. doi:[10.1016/S0262-8856\(02\)00098-7](https://doi.org/10.1016/S0262-8856(02)00098-7).
- [Mil67] Stanley Milgram. The small world problem. *Psychology today*, 2(1):60–67, 1967.
- [Mil02] R. Milo. Network motifs: Simple building blocks of complex networks. *Science*, 298(5594):824–827, oct 2002. doi:[10.1126/science.298.5594.824](https://doi.org/10.1126/science.298.5594.824).
- [MJ38] J. L. Moreno and H. H. Jennings. Statistics of social configurations. *Sociometry*, 1(3/4):342, jan 1938. doi:[10.2307/2785588](https://doi.org/10.2307/2785588).
- [MMF⁺08] Mark R. Meiss, Filippo Menczer, Santo Fortunato, Alessandro Flammini, and Alessandro Vespignani. Ranking web sites with real user traffic. In *Proceedings of the international conference on Web search and web data mining - WSDM '08*. ACM Press, 2008. doi:[10.1145/1341531.1341543](https://doi.org/10.1145/1341531.1341543).
- [Mor77] Jacob L. Moreno. *Who shall survive*. Beacon House Inc., New York, USA, 3rd edition edition, 1977.
- [MPL17] Naoki Masuda, Mason A. Porter, and Renaud Lambiotte. Random walks and diffusion on networks. *Physics Reports*, 716-717:1–58, nov 2017. doi:[10.1016/j.physrep.2017.07.007](https://doi.org/10.1016/j.physrep.2017.07.007).
- [MR97] Heikki Mannila and Pirjo Ronkainen. Similarity of event sequences. In *Proceedings of the 4th International Workshop on Temporal Representation and Reasoning (TIME)*, page 136. IEEE Computer Society, 1997. doi:[10.1109/TIME.1997.600793](https://doi.org/10.1109/TIME.1997.600793).
- [MRPG18] Letizia Milli, Giulio Rossetti, Dino Pedreschi, and Fosca Giannotti. Information diffusion in complex networks: The active/passive conundrum. In Chantal Cherifi, Hocine Cherifi, Márton Karsai, and Mirco Musolesi, editors, *Complex Networks & Their Applications VI*, pages 305–313, Cham, 2018. Springer International Publishing. doi:[10.1007/978-3-319-72150-7_25](https://doi.org/10.1007/978-3-319-72150-7_25).
- [MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
- [Nav01] Gonzalo Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, March 2001. doi:[10.1145/375360.375365](https://doi.org/10.1145/375360.375365).
- [New02] Mark E.J. Newman. The structure and function of networks. *Computer Physics Communications*, 147(1-2):40–45, aug 2002. doi:[10.1016/s0010-4655\(02\)00201-1](https://doi.org/10.1016/s0010-4655(02)00201-1).

-
- [New05] Mark E.J. Newman. A measure of betweenness centrality based on random walks. *Social networks*, 27(1):39–54, 2005. doi:[10.1016/j.socnet.2004.11.009](https://doi.org/10.1016/j.socnet.2004.11.009).
- [New10] Mark E.J. Newman. *Networks: An Introduction*. Oxford University Press, New York, USA, 2010.
- [Nie74] Juhani Nieminen. On the centrality in a graph. *Scandinavian Journal of Psychology*, 15(1):332–336, sep 1974. doi:[10.1111/j.1467-9450.1974.tb00598.x](https://doi.org/10.1111/j.1467-9450.1974.tb00598.x).
- [Nii87] I Niiniluoto. *Truthlikeness*, volume 185. D.Reidel Publishing Company, Dordrecht, Holland, 1987.
- [Nor83] Donald A. Norman. Design Rules Based on Analyses of Human Error. *Commun. ACM*, 26(4):254–258, 1983. doi:[10.1145/2163.358092](https://doi.org/10.1145/2163.358092).
- [NR04] Jae Dong Noh and Heiko Rieger. Random walks on complex networks. *Physical Review Letters*, 92(11), mar 2004. doi:[10.1103/physrevlett.92.118701](https://doi.org/10.1103/physrevlett.92.118701).
- [NS72] A. Newell and H. A. Simon. *Human Problem Solving*. Prentice Hall, Englewood Cliffs, 1972.
- [NW70] Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, 1970. doi:[10.1016/0022-2836\(70\)90057-4](https://doi.org/10.1016/0022-2836(70)90057-4).
- [Odd86] Graham Oddie. *Likeness to Truth*. D.Reidel Publishing Company, Dordrecht, Holland, 1986.
- [PBMW99] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.
- [PL05] Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. In *Computer and Information Sciences - ISCI 2005*, pages 284–293. Springer, Berlin, Heidelberg, 2005. doi:[10.1007/11569596_31](https://doi.org/10.1007/11569596_31).
- [Qui86] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986. doi:[10.1007/BF00116251](https://doi.org/10.1007/BF00116251).
- [Ras82] Jens Rasmussen. Human errors. A taxonomy for describing human malfunction in industrial installations. *Journal of occupational accidents*, 4(2-4):311–333, 1982. doi:[10.1016/0376-6349\(82\)90041-4](https://doi.org/10.1016/0376-6349(82)90041-4).
- [RB01] Jan Ramon and Maurice Bruynooghe. A polynomial time computable metric between point sets. *Acta Informatica*, 37(10):765–780, 2001.
- [RB08] Martin Rosvall and Carl T Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, 2008. doi:[10.1073/pnas.0706851105](https://doi.org/10.1073/pnas.0706851105).
- [Rea90] James Reason. *Human error*. Cambridge University Press, 1990.
- [REL⁺14] Martin Rosvall, Alcides V Esquivel, Andrea Lancichinetti, Jevin D West, and Renaud Lambiotte. Memory in network flows and its effects on spreading dynamics and community detection. *Nature communications*, 5, 2014. doi:[10.1038/ncomms5630](https://doi.org/10.1038/ncomms5630).
-

- [RLH11] Luis E. C. Rocha, Fredrik Liljeros, and Petter Holme. Simulated epidemics in an empirical spatiotemporal network of 50,185 sexual contacts. *PLoS Computational Biology*, 7(3):e1001109, mar 2011. doi:[10.1371/journal.pcbi.1001109](https://doi.org/10.1371/journal.pcbi.1001109).
- [RM08] Lior Rokach and Oded Z Maimon. *Data mining with decision trees: theory and applications*, volume 69. World scientific, 2008.
- [RMK11] Daniel M. Romero, Brendan Meeder, and Jon Kleinberg. Differences in the Mechanics of Information Diffusion Across Topics: Idioms, Political Hashtags, and Complex Contagion on Twitter. In *Proceedings of the 20th International Conference on World Wide Web, WWW '11*, pages 695–704, New York, NY, USA, 2011. ACM. doi:[10.1145/1963405.1963503](https://doi.org/10.1145/1963405.1963503).
- [RSF11] Marco Ragni, Felix Steffenhagen, and Thomas Fangmeier. A Structural Complexity Measure for Predicting Human Planning Performance. In *Proceedings of the Cognitive Science Society*, volume 33, 2011.
- [Sab66] Gert Sabidussi. The centrality index of a graph. *Psychometrika*, 31(4):581–603, 1966. doi:[10.1007/BF02289527](https://doi.org/10.1007/BF02289527).
- [SB00] N Sumpter and A Bulpitt. Learning spatio-temporal patterns for predicting object behaviour. *Image and Vision Computing*, 18(9):697–704, jun 2000. doi:[10.1016/s0262-8856\(99\)00073-6](https://doi.org/10.1016/s0262-8856(99)00073-6).
- [SBvLP⁺11] Judy Shamoun-Baranes, E Emiel van Loon, Ross S Purves, Bettina Speckmann, Daniel Weiskopf, and CJ Camphuysen. Analysis and visualization of animal movement. *Biology letters*, 2011. doi:[10.1098/rsbl.2011.0764](https://doi.org/10.1098/rsbl.2011.0764).
- [Sch17] Ingo Scholtes. When is a Network a Network? Multi-Order Graphical Model Selection in Pathways and Temporal Networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '17*, page 1037–1046, New York, NY, USA, 2017. Association for Computing Machinery. doi:[10.1145/3097983.3098145](https://doi.org/10.1145/3097983.3098145).
- [SFS⁺09] Frank Schweitzer, Giorgio Fagiolo, Didier Sornette, Fernando Vega-Redondo, Alessandro Vespignani, and Douglas R. White. Economic networks: The new challenges. *Science*, 325(5939):422–425, jul 2009. doi:[10.1126/science.1173644](https://doi.org/10.1126/science.1173644).
- [SH03] Rituparna Sen and Mark H Hansen. Predicting web users' next access based on log data. *Journal of Computational and Graphical Statistics*, 12(1):143–155, mar 2003. doi:[10.1198/1061860031275](https://doi.org/10.1198/1061860031275).
- [Shu08] Valerie J. Shute. Focus on formative feedback. *Review of Educational Research*, 78(1):153–189, 2008, <https://doi.org/10.3102/0034654307313795>. doi:[10.3102/0034654307313795](https://doi.org/10.3102/0034654307313795).
- [SIVMZN12] SR Sudarshan Iyengar, CE Veni Madhavan, Katharina A Zweig, and Abhiram Natarajan. Understanding human navigation using network analysis. *Topics in cognitive science*, 4(1):121–134, 2012. doi:[10.1111/j.1756-8765.2011.01178.x](https://doi.org/10.1111/j.1756-8765.2011.01178.x).
- [SJ10] Marcel Salathé and James H. Jones. Dynamics and control of diseases in networks with community structure. *PLoS Computational Biology*, 6(4):e1000736, apr 2010. doi:[10.1371/journal.pcbi.1000736](https://doi.org/10.1371/journal.pcbi.1000736).

- [SND10] Jan-Willem Strijbos, Susanne Narciss, and Katrin Dünnebier. Peer feedback content and sender's competence level in academic writing revision tasks: Are they critical for feedback perceptions and efficiency? *Learning and Instruction*, 20(4):291–303, aug 2010. doi:[10.1016/j.learninstruc.2009.08.008](https://doi.org/10.1016/j.learninstruc.2009.08.008).
- [Spe04] C. Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72, jan 1904. doi:[10.2307/1412159](https://doi.org/10.2307/1412159).
- [STYS16] Natsuki Sano, Reo Tsutsui, Katsutoshi Yada, and Tomomichi Suzuki. Clustering of customer shopping paths in japanese grocery stores. *Procedia Computer Science*, 96:1314–1322, 2016. doi:[10.1016/j.procs.2016.08.176](https://doi.org/10.1016/j.procs.2016.08.176).
- [SWG16] Ingo Scholtes, Nicolas Wider, and Antonios Garas. Higher-Order Aggregate Networks in the Analysis of Temporal Networks: Path structures and centralities. *European Physical Journal B*, 89(3):1–15, 2016. doi:[10.1140/epjb/e2016-60663-0](https://doi.org/10.1140/epjb/e2016-60663-0).
- [Syl78] J. J. Sylvester. Chemistry and algebra. *Nature*, 17(432):284–284, feb 1878. doi:[10.1038/017284a0](https://doi.org/10.1038/017284a0).
- [SZ89] Karen Stephenson and Marvin Zelen. Rethinking centrality: Methods and examples. *Social Networks*, 11(1):1–37, mar 1989. doi:[10.1016/0378-8733\(89\)90016-6](https://doi.org/10.1016/0378-8733(89)90016-6).
- [Tra17] Transport for London. Rolling Origin and Destination Survey (RODS), 2017. Available at <http://www.tfl.gov.uk/info-for/open-data-users/our-feeds>.
- [TS04] Alessio Toraldo and Tim Shallice. Error analysis at the level of single moves in block design. *Cognitive neuropsychology*, 21(6):645–659, 2004. doi:[10.1080/02643290342000591](https://doi.org/10.1080/02643290342000591).
- [TZ16] Sude Tavassoli and Katharina Anna Zweig. Most central or least central? How much modeling decisions influence a node's centrality ranking in multiplex networks. In *Proceedings of the third European Network Intelligence Conference (ENIC 2016)*, pages 25–32, 2016.
- [UMZ⁺12] Stefan Uhlmann, Heiko Mannsperger, Jitao David Zhang, Emöke-Ágnes Horvat, Christian Schmidt, Moritz Küblbeck, Frauke Henjes, Aoife Ward, Ulrich Tschulena, Katharina Zweig, Ulrike Korf, Stefan Wiemann, and Özgür Sahin. Global microRNA level regulation of EGFR-driven cell-cycle protein network in breast cancer. *Molecular Systems Biology*, 8(1):570, jan 2012. doi:[10.1038/msb.2011.100](https://doi.org/10.1038/msb.2011.100).
- [Ves18] Alessandro Vespignani. Twenty years of network science. *Nature*, 558(7711):528–529, jun 2018. doi:[10.1038/d41586-018-05444-y](https://doi.org/10.1038/d41586-018-05444-y).
- [VHL⁺14] Marie Vogel, Ronan Hamon, Guillaume Lozenguez, Luc Merchez, Patrice Abry, Julien Barnier, Pierre Borgnat, Patrick Flandrin, Isabelle Mallon, and Céline Robardet. From bicycle sharing system movements to users: a typology of Vélo'v cyclists in Lyon based on large-scale behavioural dataset. *Journal of Transport Geography*, 41:280–291, 2014. doi:[10.1016/j.jtrangeo.2014.07.005](https://doi.org/10.1016/j.jtrangeo.2014.07.005).
- [Vic02] Tamas Vicsek. Complexity: The bigger picture. *Nature*, 418(6894):131–131, jul 2002. doi:[10.1038/418131a](https://doi.org/10.1038/418131a).
- [VKG02] Michail Vlachos, George Kollios, and Dimitrios Gunopulos. Discovering similar multidimensional trajectories. In *Proceedings of the 18th International Conference on Data*

- Engineering*, pages 673–684. IEEE, 2002. doi:[10.1109/ICDE.2002.994784](https://doi.org/10.1109/ICDE.2002.994784).
- [WF94] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge University Press, 1994.
- [Wik07] Wikipedia. 2007 Wikipedia Selection for schools. <https://web.archive.org/web/20171022101730/http://schools-wikipedia.org/>, 2007.
- [WL12a] Robert West and Jure Leskovec. Automatic versus human navigation in information networks. In *Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media*, 2012.
- [WL12b] Robert West and Jure Leskovec. Human wayfinding in information networks. In *Proceedings of the 21st international conference on World Wide Web*, pages 619–628, New York, NY, USA, 2012. ACM. doi:<https://doi.org/10.1145/2187836.2187920>.
- [Wor] WordFind. <http://www.wordfind.com/3-letter-words>.
- [WPP09] Robert West, Joelle Pineau, and Doina Precup. Wikispeedia: An online game for inferring semantic distances between concepts. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence, IJCAI'09*, page 1598–1603, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.
- [WRP⁺13] Lilian Weng, Jacob Ratkiewicz, Nicola Perra, Bruno Gonçalves, Carlos Castillo, Francesco Bonchi, Rossano Schifanella, Filippo Menczer, and Alessandro Flammini. The role of information diffusion in the evolution of social networks. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '13*, pages 356–364, New York, NY, USA, 2013. ACM. doi:[10.1145/2487575.2487607](https://doi.org/10.1145/2487575.2487607).
- [WS98] Duncan J Watts and Steven H Strogatz. Collective dynamics of ‘small-world’ networks. *Nature*, 393(6684):440–442, 1998. doi:[10.1038/30918](https://doi.org/10.1038/30918).
- [WS03] Scott White and Padhraic Smyth. Algorithms for estimating relative importance in networks. In *Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 266–275. ACM, 2003. doi:[10.1145/956750.956782](https://doi.org/10.1145/956750.956782).
- [XWC16] Jian Xu, Thanuka L. Wickramaratne, and Nitesh V. Chawla. Representing higher-order dependencies in networks. *Science Advances*, 2(5):e1600028, may 2016. doi:[10.1126/sciadv.1600028](https://doi.org/10.1126/sciadv.1600028).
- [YML⁺14] Jaewon Yang, Julian McAuley, Jure Leskovec, Paea LePendou, and Nigam Shah. Finding progression stages in time-evolving event sequences. In *Proceedings of the 23rd international conference on World wide web - WWW '14*. ACM Press, 2014. doi:[10.1145/2566486.2568044](https://doi.org/10.1145/2566486.2568044).
- [YW03] Jiong Yang and Wei Wang. CLUSEQ: efficient and effective sequence clustering. In *Proceedings 19th International Conference on Data Engineering Cat No03CH37405*, pages 101–112, 2003. doi:[10.1109/ICDE.2003.1260785](https://doi.org/10.1109/ICDE.2003.1260785).
- [YZZ⁺10] Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang. T-drive: driving directions based on taxi trajectories. In *Proceedings of the 18th SIGSPATIAL International conference on advances in geographic information*

- systems*, pages 99–108. ACM, 2010. doi:[10.1145/1869790.1869807](https://doi.org/10.1145/1869790.1869807).
- [ZL15] Shanjiang Zhu and David Levinson. Do People Use the Shortest Path? An Empirical Test of Wardrop’s First Principle. *PLOS ONE*, 10(8):e0134322, aug 2015. doi:[10.1371/journal.pone.0134322](https://doi.org/10.1371/journal.pone.0134322).
- [ZPJS04] Jiajie Zhang, Vimla L Patel, Todd R Johnson, and Edward H Shortliffe. A cognitive taxonomy of medical errors. *Journal of Biomedical Informatics*, 37(3):193–204, 2004. doi:<https://doi.org/10.1016/j.jbi.2004.04.004>.
- [Zwe16] Katharina A. Zweig. *Network Analysis Literacy*. Springer-Verlag KG, 2016. doi:[10.1007/978-3-7091-0741-6](https://doi.org/10.1007/978-3-7091-0741-6).
- [ZYPW02] Li Zhao, Sung Sam Yuan, Sun Peng, and Ling Tok Wang. A new efficient data cleansing method. In *Database and Expert Systems Applications*, pages 484–493. Springer, 2002. doi:[10.1007/3-540-46146-9_48](https://doi.org/10.1007/3-540-46146-9_48).
- [ZZXM09] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. Mining interesting locations and travel sequences from GPS trajectories. In *Proceedings of the 18th international conference on World wide web*, pages 791–800. ACM, 2009. doi:[10.1145/1526709.1526816](https://doi.org/10.1145/1526709.1526816).

List of abbreviations and notations

Graph and walk definitions

$G = (V, E)$	graph with a V set of nodes, $E \subseteq V \times V$ set of edges
$ P $	length of a walk
$V(P)$	node set of a walk
$d(v, w)$	length of shortest path from node v to node w
$d_F(v)$	for graphs with goal states: length of shortest path from node v to closest goal state
$d(v, P)$	length of shortest path from node v to closest node in $V(P)$
$deg(v)$	degree of node v
$\mathcal{P} = \{P_1, P_2, \dots, P_\ell\}$	(multi-)set of walks (<i>trajectories</i>) in graph G
$s(P)$	source node of walk P
$t(P)$	target node of walk P
$I(P)$	index set of walk P
$P(i)$	i -th node of walk P
$P_{i,j}$	sub-path of walk P containing the nodes $P(i), P(i+1), \dots, P(j)$
$V^{\mathcal{P}}$	subset of nodes of V that are contained in at least one trajectory of \mathcal{P}
$q(\mathcal{P}_c)$	for game trajectories: fraction of solving/non-solving trajectories in \mathcal{P}_c
σ_{st}	number of shortest paths from node s to node t
$\sigma_{st}(v)$	number of shortest paths from node s to node t containing v
$\sigma_{st}^{\mathcal{P}}(v)$	flow-based version of $\sigma_{st}(v)$ (based on trajectory set \mathcal{P})
k -core	a maximal connected subgraph of a graph G in which all nodes have a degree of at least k

Further definitions

\mathbb{N}	set of natural numbers (including 0)
\mathbb{R}	set of real numbers
(X, d)	metric space: set X and distance metric d on X
μ	mapping relation

μ^*	cost-optimal mapping
$\mathcal{H}_{P,Q}$	set of all relations on $I(P) \times I(Q)$
$\Gamma = \{\gamma_1, \dots, \gamma_l\}$	grouping (clustering)
$\sigma(v)$	ranking position of node v
σ^o	ordinal ranking
σ^{min}	standard competition ranking
σ^f	fractional ranking

Measures

$s : X \times X \rightarrow \mathbb{R}$	similarity measure on X
$\delta : X \times X \rightarrow \mathbb{R}$	distance measure on X
distance metric	distance measure satisfying non-negativity, coincidence, symmetry, and the triangle inequality
D	distance measure for clusters
τ_P	Pearson correlation coefficient
τ_K	Kendall rank correlation coefficient
τ	unweighted overlap of rankings
τ_w	weighted overlap of rankings
$ov(\sigma_1, \sigma_2, x)$	number of common elements in first x positions of rankings σ_1 and σ_2
$span(v)$	span of ranking positions of node v
$D(v)$	degree centrality of node v
$C^{\rightarrow}(v)$	out-closeness centrality of node v
$C^{\leftarrow}(v)$	in-closeness centrality of node v
$B(v)$	betweenness centrality of node v
$B_e(v)$	betweenness centrality including endpoints of node v
B_S, B_{SW}, B_R, B_{RW}	flow-based betweenness variants
$C_S, C_{S'}, C_{SW}, C_R, C_{RW}, C_{RW'}$	flow-based closeness variants
Gini coefficient	measure of statistical dispersion
σ_{nss}	node set similarity
σ_{ess}	edge set similarity
σ_{lcs}	LCS similarity (based on longest common subsequence)
δ_h	Hausdorff distance
δ_{mad}	matched average distance
δ_s	surjection distance
δ_F	Fréchet distance for (continuous) curves
δ_{dF}	discrete Fréchet distance for polygonal curves
δ_{adF}	additive discrete Fréchet distance for polygonal curves

δ_{sadF}	simplified additive discrete Fréchet distance
$purity(\gamma)$	purity of a single cluster $\gamma \in \Gamma$
$purity(\Gamma)$	unweighted average purity of a clustering Γ
$purity_w(\Gamma)$	weighted average purity of a clustering Γ
$C(\mathcal{P})$	network coverage by trajectory set \mathcal{P}
$nu(v)$	node usage of node v by trajectories in \mathcal{P}
$npu(s, t)$	node pair usage for node pair $(s, t) \in V \times V$
$npu(k)$	node pair usage for graph distance k
$np(k)$	number of node pairs in graph G with graph distance k

Naming conventions

P, Q, R	variable names for walks
v, w, x	variable names for nodes
$e = (v, w)$	variable name for edges
i, j, k, l	variable names for indices (natural numbers)

Abbreviations

UNC	uniform neighbor choice
BWR	backwards-restricted neighbor choice
PL	path length restriction
LC	line change restriction
IF	Internal Feedback; also: control group only generating Internal Feedback
PF	Peer Feedback; also: experimental group generating Peer Feedback
HighRC	High Response Confidence
LowRC	Low Response Confidence
InI	Boolean flag whether an error move is contained in initial solution
InP	flag whether an error move is contained in peer solution
MAW_IF	flag whether error move in initial solution is Marked As Wrong by participant
MAW_PF	flag whether error move in peer solution is Marked As Wrong by participant
DB1B	airline origin and destination survey; identifier for dataset containing airline passenger journeys
RH	Rush Hour

Short Curriculum Vitæ

Personal Information

Mareike Bockholt
Contact mareike.bockholt@gmail.com



Education

2013 - 2015 **Master of Science** in Computer Science, Technische Universität Kaiserslautern
Master thesis *Measures for the similarity of Paths in Complex Networks*
2009 - 2013 **Bachelor of Science** in Applied Computer Science, Universität Heidelberg
Bachelor thesis *Ein netzwerkanalytischer Ansatz zur Untersuchung der Komplexität des Rush-Hour-Spiels*
1999 - 2008 **Abitur**, Julius-Stursberg-Gymnasium, Neukirchen-Vluyn

Employment

October 2015 - October 2020 Research assistant in Algorithm Accountability Lab, Department of Computer Science, Technische Universität Kaiserslautern
August 2018 - May 2019 Parental leave

Publications

- [1] Mareike Bockholt, Olaf Peters, Susanne Narciss, and Katharina A. Zweig. Analysis of human problem solving drafts: a methodological approach on the example of Rush Hour. In *Proceedings of the 40th Annual Meeting of the Cognitive Science Society*, 2018.
- [2] Mareike Bockholt and Katharina A. Zweig. Clustering of paths in complex networks. In Hocine Cherifi, Sabrina Gaito, Walter Quattrociocchi, and Alessandra Sala, editors, *Complex Networks & Their Applications V*, pages 183–195. Springer International Publishing, Cham, 2017.
- [3] Mareike Bockholt and Katharina A. Zweig. Paths in complex networks. In R. Alhajj and J. Rokne, editors, *Encyclopedia of Social Network Analysis and Mining*. Springer, 2018.
- [4] Mareike Bockholt and Katharina A. Zweig. Process-driven betweenness centrality measures. In *Lecture Notes in Social Networks*, pages 17–33. Springer International Publishing, 2018. **Best Paper Award.**
- [5] Mareike Bockholt and Katharina A. Zweig. Why we need a process-driven network analysis. In *Complex Networks and Their Applications VIII*, pages 81–93. Springer International Publishing, Cham, nov 2019.
- [6] Mareike Bockholt and Katharina A. Zweig. A systematic evaluation of assumptions in centrality measures by empirical flow data. *Social Network Analysis and Mining*, 11(1), mar 2021.
- [7] Mareike Bockholt and Katharina Anna Zweig. Why Is This So Hard? Insights from the State Space of a Simple Board Game. In Stefan Göbel, Minhua Ma, Jannicke Baalsrud Hauge, Manuel Fradinho Oliveira, Josef Wiemeyer, and Viktor Wendel, editors, *Serious Games*, pages 147–157, Cham, 2015. Springer International Publishing.
- [8] Mareike Bockholt and Katharina Anna Zweig. Towards a process-driven network analysis. *Applied Network Science*, 5(1), August 2020.
- [9] Katharina Emmerich and Mareike Bockholt. Serious games evaluation: Processes, models, and concepts. In *Entertainment Computing and Serious Games*, pages 265–283. Springer International Publishing, 2016.
- [10] Katharina Emmerich, Natalya Bogacheva, Mareike Bockholt, and Viktor Wendel. Operationalization and measurement of evaluation constructs. In *Entertainment Computing and Serious Games*, pages 306–331. Springer International Publishing, 2016.

