Adaptation of Synthesis Plans in Organic Chemistry

Jean Lieber and Amedeo Napoli[1]

Abstract. This paper presents RESYN/CBR, a case-based planner dedicated to organic chemistry synthesis, from the viewpoint of its adaptation process.

1 INTRODUCTION

Case-based reasoning [10] is a problem-solving paradigm based on the use of a *case* base where a case is a pair (problem, solution) and, in particular, a *source* case is a case from the case base. Given a problem to solve –the *target* problem– a case-based reasoning system first searches a source case *similar* to the target problem (*retrieval* task) and then adapt this *retrieved* case in order to solve the target problem (*adaptation* task). This paper describes RESYN/CBR, an application of case-based planning to organic synthesis, from the viewpoint of case adaptation. Section 2 presents an overview of RESYN/CBR and its application domain. Section 3 describes the retrieval process. The adaptation process is described in section 4. It consists in *matching* the retrieved case to the target problem and *reusing* this retrieved case in order to propose a solution to the target problem. The discussion of section 5 concludes the paper. In the appendix (section 6) the RESYN/CBR approach is compared to other approaches.

2 OVERVIEW OF SYNTHESIS PLANNING

This section presents the domain of computer-assisted synthesis in organic chemistry and an application of case-based planning in this domain.

2.1 Computer-Assisted Synthesis in Organic Chemistry

One of the main objectives of organic synthesis in chemistry is to build up molecules called *target molecules*, from *simpler* molecules called *starting materials* [5]. The way a molecule $\mathtt{m}$ is built is described by a *synthesis plan* for $\mathtt{m}$ denoted by $\mathtt{P}[\mathtt{m}]$.

Below, we introduce a few basic notions of organic synthesis planning. A molecule can be seen as a non-directed graph whose vertices represent atoms and edges represent bonds. Given two molecules $\mathtt{M}_1$ and $\mathtt{M}_2$, $\mathtt{M}_1 \Longrightarrow \mathtt{M}_2$ (or $\mathtt{M}_2 \Longleftarrow \mathtt{M}_1$) denotes a chemical *transform* and means that $\mathtt{M}_1$ can be reduced to $\mathtt{M}_2$ as in problem reduction. $\mathtt{M}_1$ is called the *data* and $\mathtt{M}_2$ the *result* of the transform. A synthesis plan $\mathtt{P}[\mathtt{m}]$ is an ordered set of transforms such that $\mathtt{M}$ is the data of the first transform. $\mathtt{M}$ is called the *head* of the plan. In figure 1, $\mathtt{M}_1$ is the head and $\mathtt{M}_4$ is the starting material (the last result of the plan).

The goal of computer-assisted synthesis (CAS) is to assist a chemist elaborating a synthesis plan for a chosen target molecule $\mathtt{m}$. RESYN [18] is a CAS system and is the basis of RESYN/CBR, a case-based planner

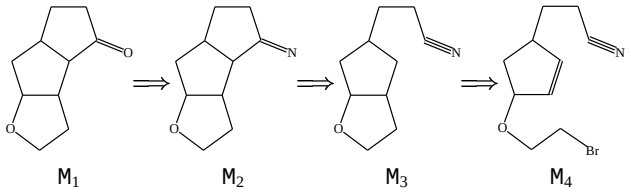

Figure 1. A synthesis plan $\mathtt{P}[\mathtt{M}_1]$ borrowed from [5]. $\mathtt{M}_1$, $\mathtt{M}_2$, $\mathtt{M}_3$ and $\mathtt{M}_4$ represent molecules. Br, O and N are atom symbols respectively denoting bromines, oxygens and nitrogens. The non-labelled vertices of the graphs represent carbons. The bonds are simple, double or triple and they are represented by simple, double and triple lines.

whose goal is to suggest a synthesis plan $\mathtt{P}[\mathtt{m}]$ for $\mathtt{m}$. In RESYN, atoms and bonds are organised in a frame inheritance hierarchy. The *co-subsumption* relation is used to compare two molecules according to their composition [13]. It can be defined as follows: $\mathtt{M}$ co-subsumes $\mathtt{m}$ (denoted $\mathtt{M} \succeq \mathtt{m}$ or $\mathtt{m} \preceq \mathtt{M}$) if there exists a subgraph isomorphism from $\mathtt{M}$ to $\mathtt{m}$ respecting the atom and bond types: the atom and bond types in $\mathtt{M}$ are more general, according to the frame inheritance hierarchy, than the atom and bond types in $\mathtt{m}$. $\succeq$ is a partial ordering that can be interpreted as a "more general than" relation: if $\mathtt{M} \succeq \mathtt{m}$, $\mathtt{M}$ is said more general than $\mathtt{m}$ and $\mathtt{m}$ is said more specific than $\mathtt{M}$. Co-subsumption is illustrated by figure 2.

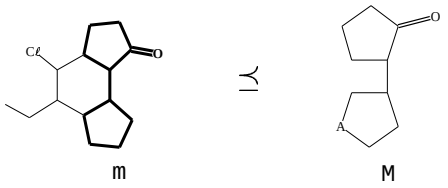

Figure 2. A co-subsumption relation between two molecules $\mathtt{M}$ and $\mathtt{m}$. The bold substructure in $\mathtt{m}$ corresponds to $\mathtt{M}$. The atom labelled as A in $\mathtt{M}$ is more general than the corresponding carbon atom in $\mathtt{m}$ (A denotes the generic atom and stands for any atom symbol).

2.2 General Description of RESYN/CBR

RESYN/CBR is a case-based planner dedicated to organic synthesis: a target problem is a target molecule, a case is a synthesis plan and, in particular, a source case is a real-world working plan. Actually, a case is a pair (planning problem, plan), but for notation simplicity we will make it similar to a plan. In [11], RESYN/CBR is described more completely with a stress on the retrieval process.

[1] CRIN CNRS – INRIA Lorraine, B.P. 239 – 54506 Vandœuvre-lès-Nancy Cedex – France (e-mail: lieber@loria.fr, napoli@loria.fr)

ECAI 96 Workshop: Adaptation in CBR.
Edited by A. Voß et al.

The case-based inference in RESYN/CBR can be considered as a sequence of three formal steps:

(1) *Retrieval* finds a source case $\mathtt{P}[\mathtt{m_k}]$ similar to the target problem $\mathtt{m}$; $\mathtt{retrieval} : \mathtt{m} \mapsto \mathtt{P}[\mathtt{m_k}]$.
(2) *Matching* relates $\mathtt{m}$ and $\mathtt{P}[\mathtt{m_k}]$. The object $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$ resulting from matching is called *similarity path*; $\mathtt{matching} : (\mathtt{P}[\mathtt{m_k}], \mathtt{m}) \mapsto \mathtt{S}[\mathtt{m_k}, \mathtt{m}]$.
(3) *Reuse* builds a solution $\mathtt{P}[\mathtt{m}]$ to the target problem $\mathtt{m}$; for that purpose, it uses the retrieved case $\mathtt{P}[\mathtt{m_k}]$ and the similarity path $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$; $\mathtt{reuse} : (\mathtt{P}[\mathtt{m_k}], \mathtt{m}, \mathtt{S}[\mathtt{m_k}, \mathtt{m}]) \mapsto \mathtt{P}[\mathtt{m}]$.

Steps (1) and (2) are implemented in a single procedure briefly presented in the next section. Step (2) is described with more details in section 4 which presents also step (3).

3 RETRIEVAL AND MATCHING

In this section the procedure performing retrieval and matching tasks is briefly explained. Let $\mathtt{m}$ be the target problem and $\mathtt{P}[\mathtt{m_k}]$ be a source case (i.e. a synthesis plan for the molecule $\mathtt{m_k}$). There are usually several possible similarity paths $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$ relating $\mathtt{m_k}$ and $\mathtt{m}$. Let $\mathtt{cost}(\mathtt{S}[\mathtt{m_k}, \mathtt{m}])$ be a numeric value associated with a similarity path $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$ ($\mathtt{cost}(\mathtt{S}[\mathtt{m_k}, \mathtt{m}]) \geq 0$). The function $\mathtt{cost}$ is used to give preference to a similarity path over other similarity paths. Let $\mathtt{d}$ be the distance defined by:

$$\mathtt{d}(\mathtt{m_k}, \mathtt{m}) = \min \{\mathtt{cost}(\mathtt{S}[\mathtt{m_k}, \mathtt{m}])\} \quad (1)$$

where $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$ is a similarity path between $\mathtt{m_k}$ and $\mathtt{m}$. This distance can be likened to an edit distance [4]. The retrieval aims at finding the plan $\mathtt{P}[\mathtt{m_k}]$ such that $\mathtt{m_k}$ is the closest to $\mathtt{m}$ according to this distance and the matching aims at finding the similarity path $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$ that corresponds to the minimal cost. Computing $\mathtt{d}(\mathtt{m_k}, \mathtt{m})$ requires finding the similarity path $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$ of lowest cost. Therefore, when the source case $\mathtt{P}[\mathtt{m_k}]$ closest to $\mathtt{m}$ is found, no more computational effort has to be spent for matching $\mathtt{m_k}$ and $\mathtt{m}$. Thus retrieval and matching are not separated in RESYN/CBR.

In practice, the procedure of retrieval and matching takes advantage of an indexing of the source cases and of a hierarchical organisation of indexes. This procedure is based on the classification processes described in [11] and is quite close to the retrieval process of MRL [8, 9].

4 ADAPTATION

In a case-based reasoning system, adaptation is usually a complex task. In RESYN/CBR, this complex task is split in simple tasks by the matching function and these simple tasks are executed by the reuse function.

4.1 Matching

Matching the retrieved case $\mathtt{P}[\mathtt{m_k}]$ and the target problem $\mathtt{m}$ consists in finding a similarity path $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$ between $\mathtt{m_k}$ and $\mathtt{m}$. A similarity path $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$ is a sequence of relations:

$$\mathtt{m_k} \preceq \mathtt{m_k}^1 \preceq \ldots \preceq \mathtt{m_k}^p \succeq \mathtt{m}^q \Longleftarrow \ldots \Longleftarrow \mathtt{m}^1 \Longleftarrow \mathtt{m} \quad (2)$$

The cost of a similarity path is established by a chemist and depends on empirical chemical data. The cost function is assumed to be additive: $\mathtt{cost}(\mathtt{S}[\mathtt{m_k}, \mathtt{m}]) = \mathtt{cost}(\mathtt{m_k} \preceq \mathtt{m_k}^1) + \ldots + \mathtt{cost}(\mathtt{m_k}^{p-1} \preceq \mathtt{m_k}^p) + \mathtt{cost}(\mathtt{m_k}^p \succeq \mathtt{m}^q) + \mathtt{cost}(\mathtt{m}^q \Longleftarrow \mathtt{m}^{q-1}) + \ldots + \mathtt{cost}(\mathtt{m}^1 \Longleftarrow \mathtt{m})$.

A similarity path is found thanks to a set of rewriting rules (actually graph rewriting rules [6]). Two types of rules are used: generalisation rules and transform rules. A generalisation rule g is a rewriting rule such that if $M \longrightarrow_g M'$ then $M \preceq M'$. A transform rule t is a rewriting rule such that if $M \longrightarrow_t M'$ then the transform $M \Longrightarrow M'$ exists.

The matching task uses an A* search [15]. In this search:

- A state is a pair $\mathtt{s} = (\mathtt{m_k} \preceq \ldots \preceq \mathtt{m_k}^p, \mathtt{m}^q \Longleftarrow \ldots \Longleftarrow \mathtt{m})$;
- The initial state is $(\mathtt{m_k}, \mathtt{m})$.
- A final state is a state $\mathtt{s}_f = (\mathtt{m_k} \preceq \ldots \preceq \mathtt{m_k}^p, \mathtt{m}^q \Longleftarrow \ldots \Longleftarrow \mathtt{m})$ such that $\mathtt{m_k}^p \succeq \mathtt{m}^q$. When a final state $\mathtt{s}_f$ is reached, a similarity path $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$ constituted by the relations of $\mathtt{s}_f$ and the relation $\mathtt{m_k}^p \succeq \mathtt{m}^q$ can be built (cf. equation (2)).
- In order to find the successors of a state $\mathtt{s}$, the system computes n_G generalisations $\mathtt{m_k}^{p+1}$ of $\mathtt{m_k}^p$ and n_T transforms $\mathtt{m}^q \Longrightarrow \mathtt{m}^{q+1}$, using as far as possible a set of available generalisation and transform rules. Thus, n_G new states of the form $(\mathtt{m_k} \preceq \ldots \preceq \mathtt{m_k}^p \preceq \mathtt{m_k}^{p+1}, \mathtt{m}^q \Longleftarrow \ldots \Longleftarrow \mathtt{m})$ and n_T new states of the form $(\mathtt{m_k} \preceq \ldots \preceq \mathtt{m_k}^p, \mathtt{m}^{q+1} \Longleftarrow \mathtt{m}^q \Longleftarrow \ldots \Longleftarrow \mathtt{m})$ are generated.
- The evaluation function is defined for a state $\mathtt{s}$ by:

$$\mathcal{E}(\mathtt{s}) = \mathtt{cost}(\mathtt{m_k} \preceq \ldots \preceq \mathtt{m_k}^p) + \mathtt{cost}(\mathtt{m}^q \Longleftarrow \ldots \Longleftarrow \mathtt{m}) + e^*(\mathtt{m_k}^p, \mathtt{m}^q) \quad (3)$$

where $e^*(\mathtt{m_k}^p, \mathtt{m}^q)$ is an estimation of the distance between $\mathtt{m_k}^p$ and $\mathtt{m}^q$.

This approach to matching is similar to the use of string matching described in [16] that is also based on an edit distance, and to structural similarity guidance presented in [3] which also uses rules to perform matching.

4.2 Reuse

Figure 3 shows a reuse process. The first column represents the retrieved synthesis plan $\mathtt{P}[\mathtt{m_k}]$. This plan is transformed into the plan $\mathtt{P}[\mathtt{m2}]$ (second column). $\mathtt{P}[\mathtt{m2}]$ is then transformed into the plan $\mathtt{P}[\mathtt{m1}]$ (third column). Finally, $\mathtt{P}[\mathtt{m1}]$ is transformed into the desired plan $\mathtt{P}[\mathtt{m}]$ which solves the target problem $\mathtt{m}$ (last column). This adaptation process in three steps is controlled by the similarity path $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$. In this example, the similarity path $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$ is represented at the first line of the figure and is constituted by the three relations $\mathtt{m_k} \preceq \mathtt{m2}$, $\mathtt{m2} \succeq \mathtt{m1}$ and $\mathtt{m1} \Longleftarrow \mathtt{m}$. The transformation of $\mathtt{P}[\mathtt{m_k}]$ into $\mathtt{P}[\mathtt{m2}]$ is based on the relation $\mathtt{m_k} \preceq \mathtt{m2}$ and is performed by a function called $\preceq$-function. $\mathtt{m_k} \preceq \mathtt{m2}$ means that $\mathtt{m2}$ is more general than $\mathtt{m_k}$; thus the $\preceq$-function executes a generalisation of the plan $\mathtt{P}[\mathtt{m_k}]$. Conversely, the transformation of $\mathtt{P}[\mathtt{m2}]$ into $\mathtt{P}[\mathtt{m1}]$ is performed by the $\succeq$-function that executes a specialisation. The transformation of $\mathtt{P}[\mathtt{m1}]$ into $\mathtt{P}[\mathtt{m}]$ is based on the relation $\mathtt{m1} \Longleftarrow \mathtt{m}$ which means that the transform $\mathtt{m} \Longrightarrow \mathtt{m1}$ exists. Actually, $\mathtt{P}[\mathtt{m}]$ is constituted by the transform $\mathtt{m} \Longrightarrow \mathtt{m1}$ and the transforms of $\mathtt{P}[\mathtt{m1}]$. Therefore the $\Longleftarrow$-function performs an extension.

More generally, the reuse process is performed by a sequence of applications of r-functions ($r \in \{\preceq, \succeq, \Longleftarrow\}$) controlled thanks to the similarity path. The r-functions are always computed in finite time. Hence, when a retrieved case $\mathtt{P}[\mathtt{m_k}]$ and a similarity path $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$ have been found, $\mathtt{P}[\mathtt{m_k}]$ is necessarily reusable for the target problem $\mathtt{m}$.

ECAI 96 Workshop: Adaptation in CBR.
Edited by A. Voß et al.

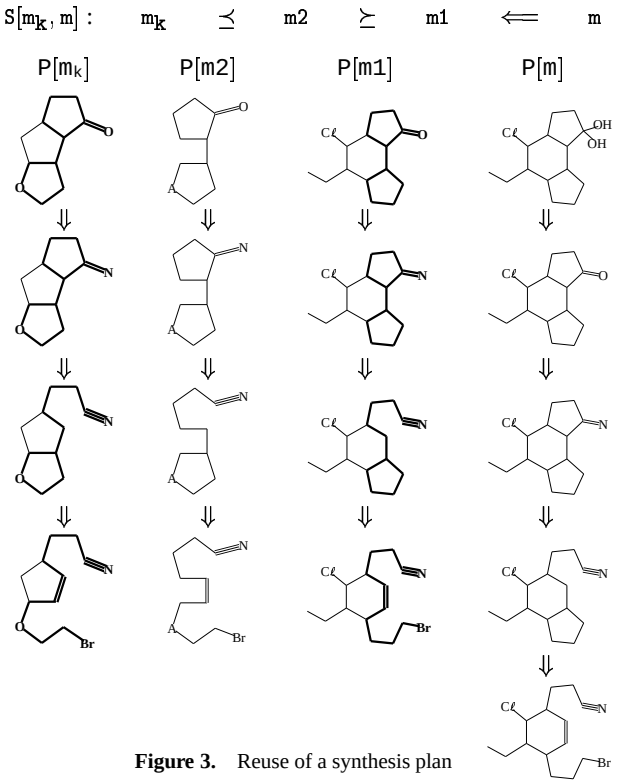

Figure 3. Reuse of a synthesis plan

5 DISCUSSION AND CONCLUSION

In this paper we have presented the adaptation process of RESYN/CBR, a case-based planner dedicated to organic synthesis. The adaptation process is performed in two steps: matching and reuse. Matching aims at relating the retrieved case and the target problem by a similarity path, i.e. a chain of relations between the retrieved case and the target problem (actually, matching is performed during retrieval). The knowledge used for matching is given by a set of generalisation and transform rules. Reuse is guided by the similarity path $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$: for each relation of $\mathtt{S}[\mathtt{m_k}, \mathtt{m}]$, an r-function computes a step of reuse. Therefore the reuse knowledge is given by the r-functions for $r \in \{\preceq, \succeq, \Longleftarrow\}$. Thus the reuse process is a sequence of generalisations, specialisations and extensions. The r-functions of RESYN/CBR correspond to the *specialists* of the multi-agent system Déjà Vu [17].

The main characteristic of the application domain of RESYN/CBR is that the problems involved are structures (i.e. graphs). This is why matching has a very important role in this system. The principle of matching in RESYN/CBR is based on the following idea: it is generally difficult to reuse directly the retrieved case to solve the target problem; thus, provided intermediate problems between the retrieved case and the target problem, the reuse task is reduced to the application of several "simple" reuse tasks. Therefore the approach to adaptation presented in this paper could be used in domains in which intermediate problems between the retrieved case and the target problem can be built in order to split the adaptation tasks into simple tasks.

6 APPENDIX

In this section we briefly compare the RESYN/CBR approach with other approaches presented at the workshop.

ECAI 96 Workshop: Adaptation in CBR.
Edited by A. Voß et al.

6.1 Abstractions and Generalisations

RESYN/CBR in its current implementation does *not* use *abstractions* but *generalisations*. These two terms are often considered as synonyms in the case-based reasoning community, but some authors make a difference. In particular, Ralph Bergmann in [1] defines abstraction as a reduction of the level of details in the description (which can entail change in the representation space, see also [2]), and generalisation as a transform along a set superset dimension. In RESYN/CBR, if $\mathtt{M} \succeq \mathtt{m}$, then $\mathtt{M}$ is more general than $\mathtt{m}$ since the set of molecules containing the substructure $\mathtt{M}$ is a superset of the set of molecules containing the substructure $\mathtt{m}$.

Among the perspectives of this work, there is the use of abstraction in RESYN/CBR. The abstraction envisaged is the one defined by Philippe Vismara in his thesis [18]: roughly said, a molecular graph is represented by another graph — called the *block representation* — which vertices correspond to the cycles and chains of the initial graph. We try to use this abstract representation of molecules (also called a *point of view* on molecules) in order to take into account the decomposition of the reasoning into a strategical level (with the block representation) and a tactical one (with the molecular representation used above). This can be likened to the use of strategies and specialists in Déjà Vu and to the use of several levels of description (and thus several levels of abstraction) presented in [12].

6.2 Structure Adaptation

In TOPO [20], maximum common subgraphs are computed in order to match the source case and the target problem. It is possible to define a set of rewriting rules that enables to compute the maximum common subgraphs thanks to the matching method presented in this paper (take for instance the generalisation rules $delete\text{-}vertex$ and $delete\text{-}edge$ applied on the target, see [4]). By contrast, when two graphs are matched thanks to a set of rewriting rules, it does not imply that this matching can be computed with maximum common subgraph calculation. Thus the RESYN/CBR approach is more general than the TOPO's one. Conversely, the RESYN/CBR's matching is probably more time consuming than the TOPO's one. In other words, TOPO and RESYN/CBR matchings correspond to different compromises between efficiency and expressiveness (the former is more efficient and the latter is more expressive). Another difference between the two approaches is that RESYN/CBR must have some domain-dependent knowledge about similarity, whereas TOPO does not need such a knowledge.

In [12], an adaptation process is seen as a sequence of substitutions on cases at different levels of description, where a case is a list of ITEMS. This description can be mapped on the reuse process described in figure 3 in the following way:

- A case is a list of ITEMS where an ITEM is a molecule.
- $\mathtt{P}[\mathtt{m_k}]$ is modified into $\mathtt{P}[\mathtt{m2}]$ by substitutions of ITEMS of $\mathtt{P}[\mathtt{m_k}]$ (substitutions by deletion of atoms and bonds and by substitution of an atom of type O by an atom of type A).
- $\mathtt{P}[\mathtt{m2}]$ is modified into $\mathtt{P}[\mathtt{m1}]$ by substitutions of ITEMS of $\mathtt{P}[\mathtt{m1}]$ (substitutions by adding atoms and bonds, and atom type substitution).
- The modification of $\mathtt{P}[\mathtt{m1}]$ into $\mathtt{P}[\mathtt{m}]$ can be seen as the substitution of $\mathtt{m1}$ by $(\mathtt{m} \Longrightarrow \mathtt{m1})$ in $\mathtt{P}[\mathtt{m1}]$.

Thus, the framework of adaptation presented in [12] can take into account the reuse process of RESYN/CBR. However, the question that remains to be answered is how the different substitution operations can

be chosen. In other words: how can the equivalent of the RESYN/CBR matching process be done?

6.3 Case Structure

In RESYN/CBR, as in Déjà Vu and PARIS [2], a case is a pair (planning problem, plan). A planning problem is represented by a molecular graph. In fact, this molecule represents only the initial state of the plan: the goal statement –*simplifying the target molecule*– is the same for all the planning problems of RESYN/CBR. A (temporal) plan is a partially ordered set of transforms $\mathtt{M}_1 \Longrightarrow \mathtt{M}_2$ (only totally ordered plans are presented in this paper).

A subplan of a given plan is itself a plan: if $m_1 \Longrightarrow m_2 \Longrightarrow m_3 \Longrightarrow m_4$ is a plan, then $m_2 \Longrightarrow m_3 \Longrightarrow m_4$ is also a plan.

In RESYN/CBR the cases and the knowledge is represented in the frame-based system Y3 [7]. As for the EADOCS [14] and INRECA [21] systems, this object-based representation has an important role in the case-based inference. Indeed, some of the generalisation rules (as the one presented in the example) use directly the frame inheritance hierarchy.

ACKNOWLEDGEMENTS

The authors would like to thank Ludmilla Mangelinck, Yannick Lallement and Arnaud Simon for their helpful comments on earlier versions of this paper. The first author would also like to acknowledge the *Institut de Recherches Servier* for its financial support.

REFERENCES

[1] R. Bergmann, 'Learning Plan Abstractions', in *GWAI-92, 16^{th} German Workshop on Artificial Intelligence*, ed., H. J. Ohlbach, Lecture Notes in Artificial Intelligence 671, 187–198, Springer Verlag, Berlin, (1992).

[2] R. Bergmann and W. Wilke, 'PARIS: FLexible Plan Adaptation by Abstraction and Refinement', In Voß [19].

[3] K. Börner, 'Structural Similarity as Guidance in Case-Based Design', in *Topics in Case-Based Reasoning – First European Workshop (EWCBR'93), Kaiserslautern*, eds., S. Wess, K.-D. Althoff, and M.M. Richter, Lecture Notes in Artificial Intelligence 837, 197–208, Springer Verlag, Berlin, (1994).

[4] H. Bunke and B. T. Messmer, 'Similarity Measures for Structured Representations', in *Topics in Case-Based Reasoning – First European Workshop (EWCBR'93), Kaiserslautern*, eds., S. Wess, K.-D. Althoff, and M.M. Richter, Lecture Notes in Artificial Intelligence 837, 106–118, Springer Verlag, Berlin, (1994).

[5] E. J. Corey and X.-M. Cheng, *The Logic of Chemical Synthesis*, John Wiley & Sons, 1989.

[6] H. Dörr, *Efficient Graph Rewriting and Its Implementation*, Lecture Notes in Computer Science 922, Springer Verlag, Berlin, 1995.

[7] R. Ducournau. Y3 : Yafool, le langage à objets et Yafen, l'interface graphique, 1991. Sema Group, Montrouge, France.

[8] J. Koehler, 'An Application of Terminological Logics to Case-based Reasoning', in *Proceedings of the 4th International Conference of Knowledge Representation and Reasoning (KR'94), Bonn*, (1994).

[9] J. Koehler, 'Planning from Second Principles', *Artificial Intelligence*, To Appear in Volume **87**, (1996).

[10] J. Kolodner, *Case-Based Reasoning*, Morgan Kaufmann, Inc., 1993.

[11] J. Lieber and A. Napoli, 'Using Classification in Case-Based Planning', in *Proceedings of the 12th European Conference on Artificial Intelligence (ECAI'96), Budapest, Hungary*, ed., W. Wahlster, pp. 132–136. John Wiley & Sons, Ltd., (1996).

[12] A. Mille, B. Fuchs, and O. Herbeaux, 'A unifying framework for Adaptation in Case-Based Reasoning', In Voß [19].

[13] A. Napoli, C. Laurenço, and R. Ducournau, 'An object-based representation system for organic synthesis planning', *International Journal of Human-Computer Studies*, **41**(1/2), 5–32, (1994).

[14] B. D. Netten and R. A. Vingerhoeds, 'Adaptation for Conceptual Desing in EADOCS', In Voß [19].

[15] J. Pearl, *Heuristics – Intelligent Search Strategies for Computer Problem Solving*, Addison-Wesley Publishing Co., Reading, MA, 1984.

[16] S. Rougegrez, 'Similarity evaluation between observed behaviours for the prediction of processes', in *Topics in Case-Based Reasoning – First European Workshop (EWCBR'93), Kaiserslautern*, eds., S. Wess, K.-D. Althoff, and M.M. Richter, Lecture Notes in Artificial Intelligence 837, pp. 155–166. Springer Verlag, Berlin, (1994).

[17] B. Smyth, 'Case Adaptation in Déjà Vu', In Voß [19].

[18] P. Vismara. Reconnaissance et représentation d'éléments structuraux pour la description d'objets complexes. Application à l'élaboration de stratégies de synthèse en chimie organique. Thèse de l'Université des Sciences et Techniques du Languedoc, Montpellier, 1995.

[19] *Proceedings of the ECAI'96 Workshop: Adaptation in Case-Based Reasoning (this volume)*, ed., A. Voß, 1996.

[20] A. Voß and C.-H. Coulon, 'Structural Adaptation with TOPO', In Voß [19].

[21] W. Wilke and R. Bergmann, 'Adaptation with the INRECA System', In Voß [19].

ECAI 96 Workshop: Adaptation in CBR.
Edited by A. Voß et al.