Controlling Nonlinear Hierarchical Planning by Case Replay*

Héctor Muñoz, Jürgen Paulokat and Stefan Wess
University of Kaiserslautern, Dept. of Computer Science
P.O. Box 3049, D-67653 Kaiserslautern, Germany
E-mail: {munioz|paulokat|wess}@informatik.uni-kl.de

Abstract

We describe a hybrid case-based reasoning system supporting process planning for machining workpieces. It integrates specialized domain dependent reasoners, a feature-based CAD system and domain independent planning. The overall architecture is build on top of CAPLAN, a partial-order nonlinear planner. To use episodic problem solving knowledge for both optimizing plan execution costs and minimizing search the case-based control component CAPLAN/CBC has been realized that allows incremental acquisition and reuse of strategical problem solving experience by storing solved problems as cases and reusing them in similar situations. For effective retrieval of cases CAPLAN/CBC combines domain-independent and domain-specific retrieval mechanisms that are based on the hierarchical domain model and problem representation.

1 Introduction

Planning for machining workpieces is a crucial step in the process chain of product development in mechanical engineering because it strongly influences the overall product's costs. However, the domain's complexity currently doesn't allow any complete analytical model so that planning must be done based on experience. The task of process planning consists of selecting and ordering machining operations such that all features of a given workpiece can be manufactured by minimal or, at least, by low costs (Cheung & Dowd, 1988). Mostly, the features of a workpiece cannot be treated independently, because steps of a manufacturing plan possibly interact. Positive interactions can be utilized to decrease the cost of manufacturing a feature, e.g. sharing preparatory steps for several processing steps, or compounding manufacturing of several features in one processing step. While positive interactions should be utilized to minimize overall production costs, negative interactions must be resolved because they lead to inconsistencies. For example a processing step for a feature must not be used if it destroys or makes it impossible to manufacture another one.

Theoretically, partial-order nonlinear planning is well suited to support the generation of process plans. Domain-independent conflict resolution techniques that are widely studied can be used for detecting negative interactions between steps. (Conflicts, threats (McAllester & Rosenblitt, 1991; Barrett & Weld, 1993) or clobbering a goal (Chapman, 1987) are used synonymously.) The truth criterion (Chapman, 1987) is an effective way of finding positive interactions. However, in most situations there are many possible treatments of interactions which one has to be chosen from. Choosing, however, is a critical step because it influences future planning and finally the execution costs of the overall plan. Past research in planning offers little support on this problem because it was mainly concerned with search for consistent solutions (e.g. (Chapman, 1987; McAllester & Rosenblitt, 1991; Barrett & Weld, 1993)) and the efficiency of the planning process itself (e.g. (Minton, 1988; Bergmann, 1992; Borrajo & Veloso, 1994)). So until now no satisfying concept is available to represent expertise adequately and support optimization of plan execution costs (Pérez & Carbonell, 1993).

A main characteristic of human planning for machining is the use of examples that have found to be successful in similar situations (Humm, Schulz, Radtke, & Warnecke, 1991). In mechanical engineering, there are numerous attempts to build up index structures to support the classification of workpieces and the

*This research was partially sponsored by the Deutsche Forschungsgemeinschaft (DFG), Sonderforschungsbereich (SFB) 314: "Künstliche Intelligenz - Wissensbasierte Systeme", Project X9 (1991 - 1995).

retrieval of associated manufacturing plans, e.g. (Optiz, 1970). However, these index structures are intended for manual use and only utilize information about the geometry, material of workpieces and the applied technology, e.g. (Zhang, Wright, & Davies, 1988). They completely lack attempts to extract structural information from plans as, e.g., in (Veloso & Carbonell, 1991) to make retrieval more informed.

In this paper, we describe CAPLAN/CBC (Humm et al., 1991; Paulokat, Praeger, & Wess, 1992) the case-based control component of the first-principle planner CAPLAN and its application in the process planning domain. It combines specialized reasoners with a general purpose planning approach (Kambhampati, Cutkosky, Tenenbaum, & Lee, 1991). For effective support of the overall reasoning process it integrates domain-independent and domain-specific methods to organize the case base and case retrieval. Like in PRODIGY/ANALOGY (Veloso, 1992) and contrary to other well-known case-based planning systems like CHEF (Hammond, 1986) case-based planning in CAPLAN/CBC means controlling the planning process of the first-principle planner by reusing control decisions of a case for solving the current problem.

The paper is organized as follows. In the next section, we describe the characteristics of the process planning domain. In section 3 we summarize the concepts of nonlinear, partial-order planning that influenced the architecture of CAPLAN/CBC. Further, we give a survey of the domain model. Section 4 describes our approach to case-based planning and in the last section we give a short discussion of our approach.

2 Domain Characteristics

The domain we are concerned with is manufacturing planning for rotary-symmetrical workpieces to be machined on a lathe. A planning problem is given as a geometrical description of a workpiece and of the raw material that only can be cylindrical in our model (Fig. 1a). The description of a workpiece is built up from geometrical primitives as cylinders, cones and toroids that describe monotone areas of the outline, possibly augmented by features as threads, grooves or special surface conditions. In most cases, the outline of a workpiece cannot be machined in one step, but repeated cutting operations are necessary to cut the difference between the raw material and the workpiece in thin horizontal or vertical layers. These layers are built up from atomic processing areas, that are automatically generated by extending the horizontal and vertical bounding lines of the geometrical primitives. Cutting an atomic area can be seen as an elementary cutting step.

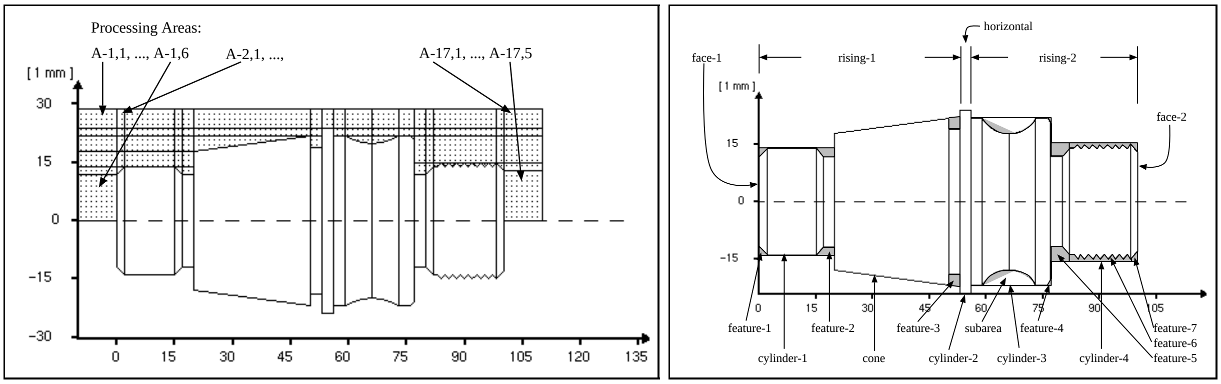

Figure 1: (a) Atomic processing areas (b) Complex processing areas of a workpiece

For machining the workpiece is clamped by a rotating clamping tool, while layers of material are removed by moving a cutting tool along the surface. Standard tools that are normally used for machining large areas have a fixed working direction, i.e., they can be used to cut off material only when being moved either to the left or to the right. Clamping a workpiece hides parts of its surface, therefore, after machining the part not hidden, it must be turned and clamped on the other side. Additionally, caused by the geometry of standard cutting tools only horizontal outlines or outlines that are rising along a tool's moving direction can be machined. To machine a descending outline requires tool changes which increase the overall production costs. But after turning a workpiece and clamping it from the other side, e.g. necessary for machining an area hidden by a clamping tool, a descending outline is a rising one and can be machined with the same tool. For finding maximally monotonously rising areas, geometrical primitives are grouped to rising areas at both ends of the workpiece and a horizontal area between them (Fig. 1b). Each of these compound areas can contain subareas that break the monotonous course of the outline. But from an abstract point of view they do not influence the construction because the monotonous outline necessarily has to be machined first before hidden subareas can be machined. For machining, the horizontal part can be added consistently to one of both rising areas, but choosing an alternative is one of the tasks of the planner because this choice can influence the plan's execution costs. Compound areas can be divided into geometric primitives, e.g. the horizontal area in Fig. 1b.

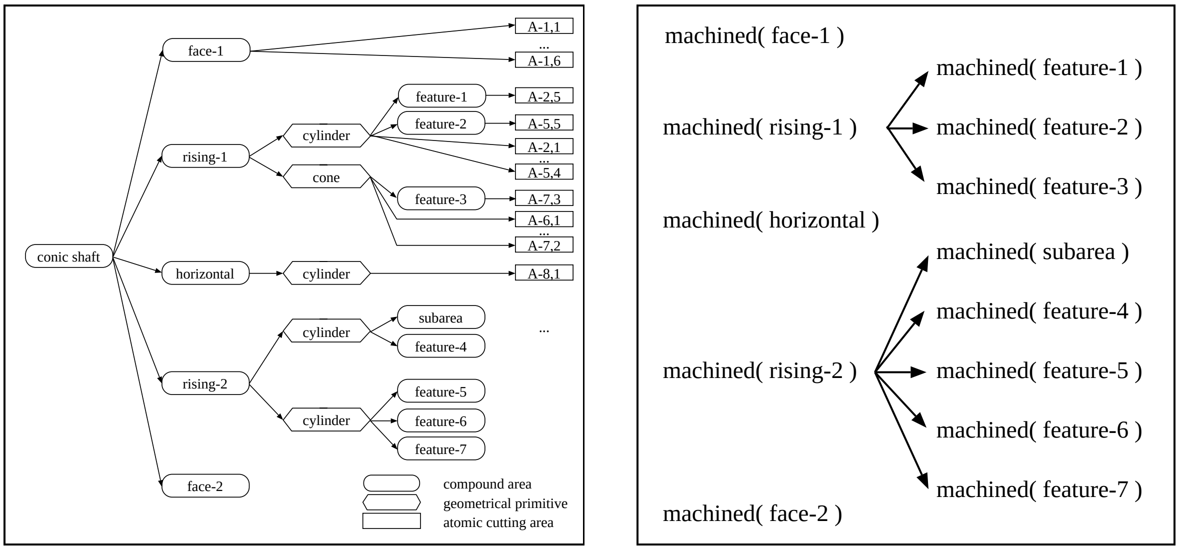

Figure 2: (a) Hierarchical representation of a planning problem (b) Set of initial goals and their ordering relations.

These primitive areas can be seen as an abstract description of the workpiece resulting in a hierarchical representation whose root node represents the whole workpiece with the compound areas as successors (Fig. 2a). Their successors are the geometrical primitives and the subareas of the workpiece. A geometrical primitive has successor nodes for its features and the atomic processing areas that are located above it. Subareas, again, can be hierarchically structured, defining their own subtrees. This hierarchical representation of the workpiece is the base for a hierarchically structured planning process.

3 Planning in CAPLAN

On an abstract level a planning problem is given by an initial situation, a set of goals, and a set of possible steps. The task of planning is computing a sequence of steps so that their execution starting from the initial situation results in a situation satisfying all goals. Planning in CAPLAN is based on the ideas of systematical non-linear planning (SNLP (McAllester & Rosenblitt, 1991)) and works on a set of partially ordered steps. A new planning problem is represented by two steps s_0 and s_∞. The effects of step s_0 are the features that are valid in the initial situation and the preconditions of s_∞ represent the features that are the goals of the planning problem. Step s_0 has no preconditions and is ordered before all other steps of the plan. Step s_∞ has no effects and is ordered behind all other steps. A goal can be satisfied by every step which is not ordered behind the precondition's step and which has an effect that can be matched with the goal's feature. This can be a step already being contained in the plan or newly added to the plan. Both, the step that adds the effect and the step the precondition of which is satisfied, are connected by a causal link that is annotated by the feature. A causal link is threated by a step if it adds an effect or the negation of an effect that can be unified with the feature of the causal link. Threats must be resolved which can be done by ordering the step that threats the causal link before the step that adds the effect or behind the step whose precondition has been satisfied or by adding constraints that make impossible the unification of effects.

Planning for machining a workpiece is preceded by transforming the set of compound areas and the set of

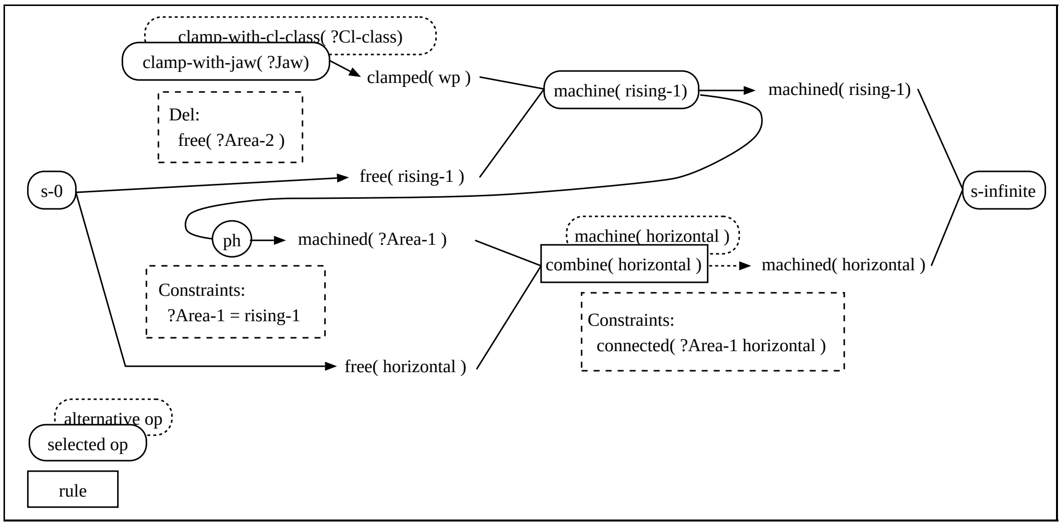

Figure 3: Partial plan on the abstract planning level.

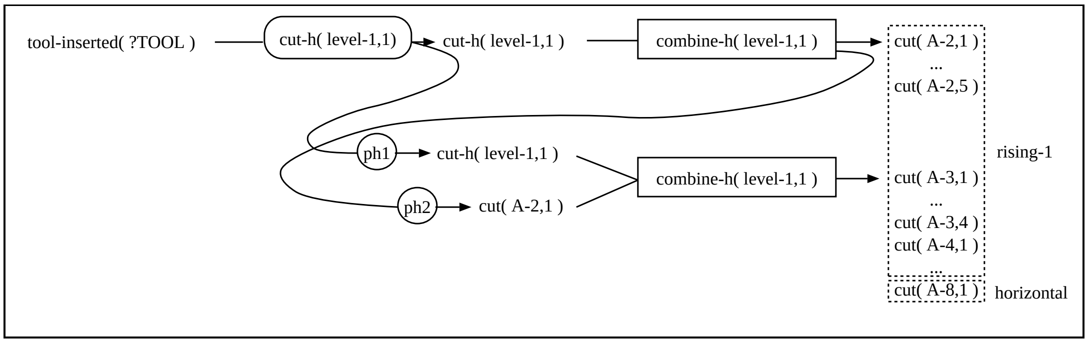

Figure 4: Partial plan on the concrete planning level for the abstract operator machine(rising-1) of Fig. 3.

features of the hierarchical description of the workpiece (Fig. 2a) in a problem representation suitable for the planner (Paulokat & Wess, 1994). Names, as rising-1 used in the problem description, provide a link to the geometrical representation. They can be used to access further information not explicitly represented in the problem description but which are necessary for control decisions. Fig. 3 shows a part of the plan on the abstract planning level of the hierarchical planning process. First, the planner decided to work on goal machined(rising-1). Therefore, the abstract operator machine(rising-1) is selected, which introduces subgoals to clamp the workpiece and to force the area rising-1 to be free. Then, planning continues with the goal machined(horizontal). Here, the planner can choose between introducing a new step to machine the horizontal component isolated from any other compound component or to utilize the side effects of another plan step. In this example the side effect of step machine(rising-1) is used. This is supported by the rule combine(horizontal) that introduces a goal that a connected area is machined. (Note: these rules are different from control rules; they are comparable to operators but do not represent actions and are not part of the resulting process plan.) The new goal machined(?Area-1) can be matched to one of the goals of the initial problem description and can be satisfied by a phantomization. As different matches are possible, this is a decision point that can strongly influence the execution costs of the plan.

By this phantomization, a decision is made that the compound areas rising-1 and horizontal are processed by one step. This influences the expansion of the abstract plan step machine(rising-1) into a new planning problem on the concrete planning level (Fig. 4). Depending on this decision, the new planning problem is to select and to order cutting operations for the atomic processing areas of the compound areas rising-1 and horizontal. Note that changing the abstraction level means changing the representation language, e.g., the complex processing area rising-1 is replaced by the atomic processing areas A-2,1, ..., A-7,2.

4 Case-Based Planning

Three characteristics make manufacturing planning a hard problem. 1) The production cost of a workpiece should be minimized. Optimization of plan execution costs, however, is not supported by AI planning techniques as described in last section. Nevertheless, they are attractive because they allow an explicit representation of plan steps and reasoning about their interactions and effects. 2) The search space for a manufacturing plan is very large which makes it impossible to generate all alternative solutions and select the best of them. And 3), there is no analytical model of the dependencies between the features of a workpiece, the possible manufacturing steps and the overall execution costs that could be used for choosing between alternatives during the planning process. Instead, human manufacturing planning is based on experience.

If no control knowledge is available, planning in CAPLAN is done by depth-first search or guided by user interactions. For support of planning in complex domains, the planning process can be controlled by a case, e.g., as in (Veloso, 1992). If a case is available the reuse of a decision that has shown to be successful in a similar situation is preferred to search when the planner has to choose at a choice point. If the rationals of a reused decision are satisfied there is a justification that it reduces backtracking and results in a better solution. In CAPLAN there are two kinds of choice points:

- the set of alternative steps to reach a goal, and
- the set of alternative constraints that can be added to a plan to resolve a threated causal link (McAllester & Rosenblitt, 1991; Barrett & Weld, 1993).

An example for a choice point where the planner has to choose to reach a goal is clamped(wp) in Figure 3. There, the conflict set consists of the steps clamp-with-jaw(?Jaw) and clamp-with-cl-class(?Cl-class).

In CAPLAN the ruse of episodic problem solving experience is done by CAPLAN/CBC. It organizes the case memory and supports case-based planning by retrieving the most similar case and by stepwise reusing the decisions stored in the case. After a problem has been successfully solved and if its solution is substantially different from that of the case it can be added to the case base. An important source for new problem solving experience are user interactions. Storing them as a new case adds control knowledge to the system and increases its planning expertise because knowledge used in these interactions possibly complements the insufficiencies of the model. Adding a case to the case base is preceded by an analyzing step that extracts the relevant features from the problem description which are used for the organization of the case base and for efficient case retrieval.

4.1 Case Storage

The organization of the case memory highly influences the efficiency and possibly the result of case retrieval. In (Veloso, 1992) a domain independent architecture for automatic case storage has been described that is based on the assumptions that the set of goals of different problem descriptions are highly varying and that the solution of a problem can be decomposed into independent subplans. These assumptions are not valid in our domain. Machining plans for the considered class of workpieces are highly sequential so that a decomposition is impossible. Further, the goals of planning problems are always similar, i.e., we always have to plan processing steps for one or two rising areas, a horizontal area and for the faces of the workpieces. Optionally, there can be a varying number of workpiece features. On the other hand, there are technological constraints that can be used to classify workpieces and, more important, that are reflected by the manufacturing plan, e.g., the kind of usable clamps depends on the ratio of the diameter of a workpiece to its length.

In our approach, these constraints are compiled in an initial, domain-specific structure of the case memory that spawns a decision tree. To add a new case to the case base its relevant category is determined by traversing the decision tree starting from its root to a leaf node. So, every new case is stored as a successor of a leaf node of the domain-specific decision tree which structures the case base. Further, the addition of a new case is preceded by footprinting (Veloso, 1992) the set of features of the initial state as a function of the goal statement and of the particular solution encountered. This process identifies the relevant features of the initial situation with respect to a plan and filters out those features that are not relevant for the plan.

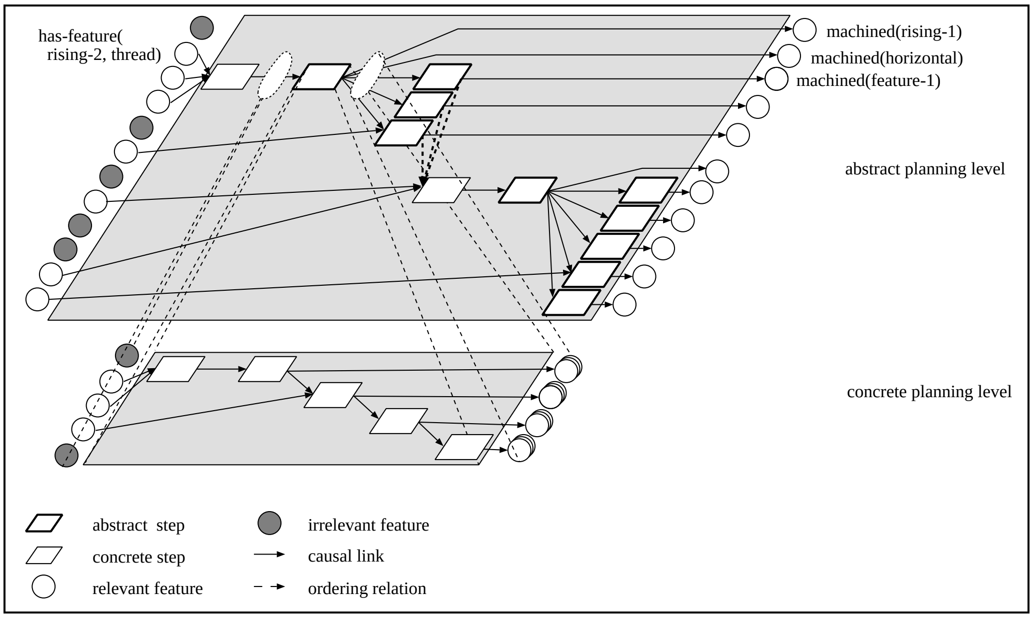

Figure 5: Interactions from a nonlinear hierarchical plan

Figure 5 shows the structure of a machining plan for the example of section 2. The footprint is independently computed for the abstract and for every concrete plan. In contrast to the approach in (Veloso, 1992) we don't use the set of goals to index a case, but the hierarchical problem representation as described in Fig. 2a. It compounds the set of goals and all features of the initial situation. This structure is annotated by the information on the relevance of a feature.

4.2 Case Retrieval

The purpose of the retrieval phase is to select a case from the case base that is useful to solve a new planning problem given. A good criterion for judging the utility of cases are the modification costs of the corresponding solutions. Unfortunately it is usually impossible to determine this measure without processing the modification (Paulokat et al., 1992; Smyth & Keane, 1993). To overcome this problem of CBR approaches domain dependent heuristics and domain independent structural methods (like footprinting (Veloso & Carbonell, 1991)) have been developed. Retrieval in CAPLAN/CBC is a combination of both approaches. The retrieval process in CAPLAN/CBC consists of two methods:

- a domain dependent pre-selection step using physical constraints and general domain knowledge to obtain a small subset of cases, discarting most of the cases in the case base.
- a footprint like (Veloso & Carbonell, 1991) domain independent selection step using structural information about previous generated plans and known problem descriptions (cf. section 4.1).

Given a CAD representation of a new workpiece (Fig. 1a) the atomic and the complex processing areas (Fig. 1b) are determined. In the next step the hierarchical representation of the planning problem (Fig. 2a) and the set of initial goals and their ordering relations (Fig. 2b) are calculated. The initial goals and ordering relations represent a kind of domain dependent technical constraints which must be satisfied in any case, e.g. the groove feature-2 in Fig. 1a can only be processed after the processing areas of cylinder-1 and the complex processing area rising-1 have been processed.

In the current implementation, case retrieval in CAPLAN/CBC is made by comparing this hierarchical description (Fig. 2a) of a new problem with the problem descriptions of stored cases (cf. Déjà Vu (Smyth & Cunningham, 1992)). Using the tree-based problem representation the case match is realized by a tree matching algorithm (Tanaka & Tanaka, 1988). The similarity measure is computed as weighted sum of all adding and deleting operations necessary for transforming the hierarchical problem description of the case into the description of the current problem. Every transformation step is associated with domain specific costs which are comparable to the effort which is necessary for the modification of the corresponding solutions (Humm et al., 1991).

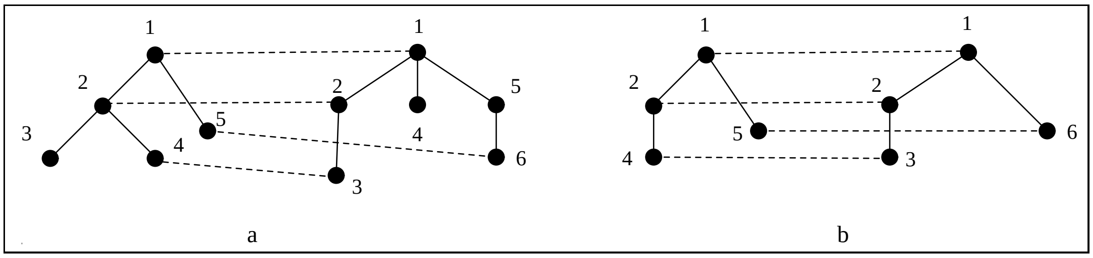

Figure 6: Mapping representations using a structural matching algorithm (a) and generating a reusable solution by deleting the respective nodes (b)

The distance between the case and the problem description is measured as follows: First, a *structural preserving match* (Tanaka & Tanaka, 1988) between the hierarchical structures of the workpieces is processed (Fig. 6a). Second, the difference between the two representations is calculated, i.e. the nodes that were not matched and the nodes that were matched, but that are of different type (for example one is a torroid and its match is a cylinder) are counted (Fig. 6b). During this calculation different domain specific weights according to the type of the nodes are processed. Since it is easier to replace an atomic processing area than a complex processing area e.g. counting a node which represents such an primitive area has an smaller value than a complex processing area. A case qualifies for solving a new problem if the computed transformation costs are less than a given limit.

To determine the relevance of features (Janetzko, Wess, & Melis, 1993) we compare the problem description with the annotated problem description (cf. section 4.1) of the case to determine in detail how features they have in common. This step is similar than using the footprinting in PRODIGY/ANALOGY (Veloso & Carbonell, 1991). Concretely, the groups of features in the annotated problem description that are completely included in the problem description are determined. The relevance of these features is defined as the number of features that are entirerly included in the problem description (Janetzko et al., 1993).

4.3 Case Replay

Although CAPLAN can autonomously solve problems by depth-first search, it is mainly intended as a planning assistant that provides a control interface for a human planner or external control components, such as CAPLAN/CBC. The input for the case-based control component consists of the problem description of a new planning problem, the case and an association between goals of the new planning problem and initial goals of the plan canned by the case. The associations are computed by tree matching of the retrieval phase. For solving a new problem, the decisions and their rationals of a case are used to choose from the set of alternative planning steps to satisfy a goal or to choose from the set of alternative constraints that can be added to resolve a threat of a causal link. This replaying of a decision of the case is done in four steps. First, the planner computes the set of alternative plan steps or conflict solving constraints that can be applied to the goal or to the conflict currently being worked on. In the second step a match of the alternative chosen by the decision of the case with the possible alternatives of the new problem is made. If there is a successful match, the decision's rationals are verified in the context of the new problem. Third, if the rationals are satisfied the step is chosen and the decision for choosing this alternative is provided with the verified rationals of the case. And fourth, the new subgoals resulting from the choosing the step are matched against the subgoals resulting from the replayed decision. By every successful match a new association between a goal of the new planning problem and a goal of the case is generated. As soon as the planner continues planning for one of matched subgoals the associations to the case are already available.

Planning for goals that cannot be matched to a goal of the case are delayed until replay is completed. The remaining goals are solved by first-principle planning or possibly by user interactions. Because replaying a case and planning by first-principles results in the same representation of a plan, all planning decisions made during case replay can be rejected if they don't allow a completion of a partial plan. The separation of the replay phase from solving unmatched goals is possible because of the partially ordered plan representation and the nonlinear planning paradigm. The first point is in contrast to PRODIGY/ANALOGY where a total-order planner is used which makes necessary interleaving of case replay and first-principle planning for unmatched goals.

5 Conclusions

We have described the hybrid planning architecture CAPLAN for supporting process planning for machining workpieces. A main characteristic of this domain is that planning is driven by minimizing plan execution costs. Human planning experts in this domain use a large number of heuristics and domain specific reasoning. Since this knowledge is extremely case sensitive its usage is not sufficiently supported by known planning techniques.

In our approach a general purpose planning system CAPLAN is combined with a case-driven control unit CAPLAN/CBC which allows to use previous experience in form of recorded cases to guide the problem solving process. If there exists a case which is useful to solve the current problem, its decisions are replayed (Veloso, 1992) in the current context, i.e., a planning decision of the case is only reused if its rationals are compatible with the problem description of the new problem. Thus, our approach is very similar to the work of Veloso (Veloso, 1992), but there are some important differences. First, planning in CAPLAN is based on the ideas of systematical nonlinear planning (SNLP (McAllester & Rosenblitt, 1991)), so the underlying planning architecture is different. Second, retrieval of useful cases in CAPLAN is realized as a combination of domain independent and domain dependent techniques which constrain the number of cases that have to be inspected during the retrieval phase. The annotated problem description is an extension of the concepts of footprinting (Veloso & Carbonell, 1991) and of interacting goals in PRODIGY. This extension is necessary due to the hierarchical representation of plans in CAPLAN. Third, the overall architecture of the system combines specialized domain dependent reasoners and a feature-based CAD system with general purpose planning (Kambhampati et al., 1991).

Contrary to other well-known case-based planning systems such as CHEF (Hammond, 1986), CAPLAN uses (like PRODIGY) an explicit domain model to describe the plan steps and the constraints that must be satisfied during the retrieval and adaptation phases. The use of domain specific knowledge and techniques for the retrieval of useful cases by calculating a *measure of adaptability* in CAPLAN is similar to the *adaptation guided retrieval* approach in Déjà Vu (Smyth & Keane, 1993).

In conclusion, nonlinear hierarchical planning is a major improvement towards solving a wider range of real-world planning problems. But a huge number of choice-points are still present, and taking wrong decisions can be very expensive. We have shown, that previous problem solving experience can be used to represent specific case sensitive knowledge and thus support the selection among these alternatives. As a result, the quality of plans and the efficiency of the planning process can be improved.

Reference

Barrett, A., & Weld, D. S. (1993). Partial-order planning. *Artificial Intelligence, 67.*

Bergmann, R. (1992). Learning abstract plans to speed up hierarchical planning. In Tadepalli, P. (Ed.), *Proceedings of Workshop "Knowledge Compilation and Speedup Learning" at the Machine Learning Conference* Scotland. University of Aberdeen.

Borrajo, D., & Veloso, M. (1994). Incremental learning of control knowledge for nonlinear problem solving. In *Proceedings of the European Conference on Machine Learning*.

Chapman, D. (1987). Planning for Conjunctive Goals. *Artificial Intelligence*, *32*, 333–377.

Cheung, Y., & Dowd, A. L. (1988). Artifical intelligence in process planning. *Computer Aided Engineering*, *5*(4), 153–156.

Hammond, K. J. (1986). *Case-Based Planning: An Integrated Theory of Planning, Learning and Memory*. Ph.D. thesis, Yale University.

Humm, B., Schulz, C., Radtke, M., & Warnecke, G. (1991). A System for Case-Based Process Planning. In *Proceedings of the 1st CIRP Workshop on Learning in Intelligent Manufacturing Systems (IMS)*. CIRP. Budapest, Hungary.

Janetzko, D., Wess, S., & Melis, E. (1993). Goal-Driven Similarity Assessment. In Ohlbach, H.-J. (Ed.), *GWAI-92: Advances in Artificial Intelligence*, pp. 283–298. Springer Verlag.

Kambhampati, S., Cutkosky, M., Tenenbaum, M., & Lee, S. H. (1991). Combining specialized reasoners and general purpose planners: A case study. In *Proceedings of AAAI-91* Menlo Park, California. MIT Press.

McAllester, D., & Rosenblitt, D. (1991). Systematic nonlinear planning. In *Proceedings of AAAI-91* Menlo Park, California. MIT Press.

Minton, S. (1988). *Learning Search Control Knowledge — An Explanation-Based Approach*. Kluwer Academic Publishers.

Optiz, H. (1970). *A Classification System to Describe Work Pieces*. Pergamon Press, Elmsford, N.Y.

Paulokat, J., Praeger, R., & Wess, S. (1992). CAbPlan – fallbasierte Arbeitsplanung. In Messer, T., & Winklhofer, A. (Eds.), *Proceedings zum 6. Workshop Planen und Konfigurieren*, pp. 169–176 Germany.

Paulokat, J., & Wess, S. (1994). Planning for machining workpieces with a partial-order, nonlinear planner. In Gil, Y., & Veloso, M. (Eds.), *AAAI 1994 Fall Symposium Series - Planning and Learning: On to real Applications*.

Pérez, M. A., & Carbonell, J. (1993). Automated acquisition of control knowledge to improve the quality of plans. Tech. rep. CMU-CS-93-142, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213.

Smyth, B., & Cunningham, P. (1992). Déjà vu: A hierarchical case-based reasoning system for software design. In Neumann, B. (Ed.), *Proceedings European Conference on AI (ECAI)*, pp. 587–589.

Smyth, B., & Keane, M. T. (1993). Retrieving adaptable cases: The role of adaptation knowledge in case retrieval. In Richter, M., Wess, S., Althoff, K., & Maurer, F. (Eds.), *Proceedings First European Workshop on Case-Based Reasoning*, pp. 76–82.

Tanaka, E., & Tanaka, K. (1988). The tree-to-tree editing problem. *International Journal of Pattern Recognition and Artificial Intelligence*, *2*.

Veloso, M., & Carbonell, J. (1991). Variable-Precision Case-Retrieval in Analogical Problem Solving. In Bareiss, R. (Ed.), *Proceedings: Case-Based Reasoning Workshop*, pp. 93–106. Morgan Kaufmann Publishers.

Veloso, M. M. (1992). *Learning by Analogical Reasoning in General Problem Solving*. CMU-CS-92-174, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213.

Zhang, K. F., Wright, A. J., & Davies, B. J. (1988). A feature-recognition knowledge base for process planning of rotational mechanical components. *International Journal of Adv. Manufacturing Technology*, *4*, 13–25.