HIERARCHICAL KNOWLEDGE-BASED PROCESS PLANNING IN MANUFACTURING

Ferenc Deák
Budapest University of Technology and Economics
dfery@bigmac.eik.bme.hu

András Kovács
Budapest University of Technology and Economics
csandris@sch.bme.hu

József Váncza
Computer and Automation Research Institute, Hungarian Academy of Sciences
vancza@sztaki.hu

Tadeusz Dobrowiecki
Budapest University of Technology and Economics
dobrowiecki@mit.bme.hu

Abstract Artificial intelligence planning methods haven't been used until recently to address the problem of computer-aided process planning (CAPP) in manufacturing in its entirety. They were simply not developed enough to tackle real-world problems of that complexity. In the paper we show that with so-called Hierarchical Task Networks, a recently matured general-purpose domain-independent planning method, we could model the planning process itself, represent and utilize different kinds of technological knowledge and keep in check the complexity of the plan generation process. To this aim the planner was extended with search methods for finding the best plans and supporting mixed-initiative, interactive planning. The proposed CAPP system deals with geometry analysis, setup planning, selection and ordering of machining operations and the assignment of resources. The first experiments with prismatic and rotational parts show considerable merit of the approach.

Keywords: Computer-aided process planning, artificial intelligence, hierarchical task network

1. INTRODUCTION

Planning of manufacturing processes provides the link between design and production. Its task is to determine a plan of discrete manufacturing operations that, when executed in an actual production environment, will

produce the part as required by its design description. Computer-Aided Process Planning (CAPP) may result in better designs, lower production costs, larger flexibility, improved quality and higher productivity.

Over the last three decades vast efforts have been made in developing novel methods and architectures for CAPP systems. The last twenty years of research has been dominated by the application of artificial intelligence (AI) methods and tools. Oddly enough, results of planning - a mainstream AI research area - found only scattered and rather simple applications in the domain of manufacturing. The reasons of this mismatch are twofold:

- CAPP is a complex problem that includes part analysis, selection of operations and resources, operation sequencing, setup planning, fixture design, and the determination of process parameters. Hence, the domain knowledge of a process planner has to cover geometry and tolerances, material properties, manufacturing processes and tools, fixtures, as well as machine tools. Besides generating executable plans, the optimal allocation of resources is the main concern of planning.
- General-purpose AI planning systems provided clear-cut logic-based representation formalisms and more and more efficient solution methods. However, the restricted representation formalisms did not allow to capture all of the relevant domain knowledge and to define planning strategies. Solvers could not handle optimization objectives and support mixed-initiative, interactive problem solving. Hence, they could not fit the real-world problems like the CAPP problem.

The goal of our research was to show that despite all the above difficulties the CAPP problem could be approached and solved as a planning problem. With the application of a recently matured general-purpose domain-independent planning method, the so-called Hierarchical Task Networks (HTN, [10]), we could model the planning process itself, represent and utilize various kinds of technological knowledge and keep also in check the complexity of the plan generation process.

In the paper a short analysis of the CAPP problem is presented. The adopted perspective is the complexity of the information management, the variety and the volume of information involved and the demands it presents for the automation and interactivity of the planning process. Next we show how the general-purpose, public domain planning methods should and can be extended to tailor them into efficient CAPP tools. The newly developed algorithms are presented in detail. The proposed approach is experimentally verified for a set of prismatic and rotational parts manufactured on vertical machine with a configurable set of tools. Finally, we discuss future extensions and the integration of the proposed methods.

2. THE CAPP PROBLEM

The *problem of CAPP* in the domain of manufacturing can be stated as follows: given (1) the descriptions of the blank part and of the finished part (in terms of geometry, dimensions, tolerances, material, and quantity), (2) the available production resources (machine tools, fixtures and tools), (3) the technological knowledge and (4) some optimization objective, find an executable and close-to-optimal plan [4,16].

There are two main approaches to process planning, the variant and the generative one. *Variant* methods are based on the retrieval and the (manual) adaptation of previous plans; they can be supported by database retrieval, and as recently, by case-based reasoning. *Generative* planners synthesize process plans. They almost unanimously depart from the geometrical CAD model of the part and work with descriptions enhanced by manufacturing *features*. Features (like e.g. holes, slots, pockets, etc.) tie together frequently occurring sub-problems with their corresponding solution patterns, i.e., geometry and tolerances with particular production methods and resources [16]. Features realize micro-worlds with both design and manufacturing related information [13]. They decompose the problem and make it amenable to efficient and automated problem solving. However, decomposing the problem is not sufficient for conquering it. Features - or rather the operations that produce them - often interact, hence the selection and merging of the appropriate plan fragments is not that straightforward. Further on, planning has to account for global technological requirements (concerning, e.g., fixturing), and overall optimization objectives as well.

The *process plan* describes how to produce the part by using the available resources. It specifies the *operations* and their *resources* (tools, fixtures and machines), the *sequence* of operations and the groups of operations - so-called *setups* - that will be performed together, by using some common resource(s). Every setup begins with mounting the part to the machine tool. The orientation of the part on the machine tool determines which operations can be executed. For some operations the orientation of the part must be changed. Hence, the setup is changed: the part is mounted to the machine tool (or to an other machine tool) in a new position.

The *planning process* can be usually considered as the hierarchy of:

1) *Setup planning*: The determination and sequencing of setups and the selection of machine tools and fixtures.
2) *Operation sequencing*: The determination and sequencing of operations and the selection of tools.
3) *Operation planning*: Determination of the machining parameters (cutting speed, feed rate etc.) and the trajectories of the tools.

There are many commercial tools to aid the last step. However, the

first two steps are very hard because the knowledge provided by the experts is usually fragmentary and inconsistent. In computer aided systems such contradictions must be solved by the human experts. Nowadays the manufacturing process is more or less automatic, but process planning still requires much work of qualified personnel. Hence, process planning is the "bottleneck" of the production [8].

3. ARTIFICIAL INTELLIGENCE AND PLANNING

Although artificial intelligence is by definition involved in the design of intelligent systems, the question of intelligence is in process planning not that important. The reason why the working knowledge and the application of the AI methods is nevertheless essential, even if no intelligent system is actually being built, is the fact that the scaling up of the real-life manufacturing problems leads to a vast body of heterogeneous, uncertain and inherently inconsistent information. In the traditional system design approach these are the major obstacles, which prohibit the usual formal specification and verification of designs.

In artificial intelligence, on the contrary, exponential complexity, uncertainty, inconsistency, interaction of various kinds of knowledge are treated as inherent attributes of complex real-life problems and numerous methods of representation and manipulation are designed to handle them [13]. Furthermore, no other methodology provides tools to tackle complicated sub-problems and to integrate the solutions into a whole.

Planning, i.e., generating sequences of actions to perform tasks and achieve objectives belongs to the core paradigms of AI [19,20]. It embraces a considerable body of abstractions, methods, and tools applicable at, and even spanning (as in hierarchical planning) many levels of abstractions. Although the paradigm of planning is one of the oldest in the AI, it is also that paradigm where the creative mixing of the ideas took and continuously takes place (consider e.g., partial ordering, least commitment strategy, using logic within a non-logical framework, anytime planning, monitoring and re-planning, planning with imperfect information, probabilistic planning, hierarchical planning, etc.).

Planning is also considered one of the most important paradigms in the future AI, serving as a kernel method for information gathering and sharing, setting up co-operation, maintaining dialogue protocols, etc. in loosely coupled intelligent systems. As a consequence, we can expect a constant flux of new developments to be tried also in the process planning domain.

General purpose, symbolic *planning methods* of mainstream AI offer in some sense a wider, in another sense narrower conceptual background than actually needed for process planning. So-called classical planners [13,19]

provide a formal language for describing the initial and goal states as well as the possible actions that may change the states. They have a plain and single measure of plan performance (plan length) and give no means for expressing control knowledge in an explicit way. They are weak in geometric reasoning and require a consistent world model [9]. The efficient generation of plans calls either for specialized methods, or for the translation of the problem case into a general combinatorial problem. Hence, such planners can hardly support mixed-initiative problem solving. This approach is also called primitive-action planning because plans are constructed solely by making use of the descriptions of the actions that may appear in the final plan [20]. Until recently, classical planners could be applied only to simple CAPP problems.

A powerful tool to overcome problems stemming from heterogeneity and complexity is the hierarchy of abstractions. Hierarchical approach, by processing information confined to the hierarchy levels and by passing information between them, makes it possible to maintain descriptions of much more complicated real systems and activities. Viewing intelligent activity as a hierarchy of tasks performed by agents and requiring expertise and resources at several levels served as the basis of the recently developed KADS knowledge modeling approach and similar efforts worldwide [15]. It came up also in planning, in various forms of so-called Hierarchical Task Network planners [10], which are now considered the most appropriate vehicle for solving real-world planning problems [20].

HTN planners generate plans that accomplish so-called *task networks.* A task network is a set of tasks to be carried out, together with constraints on the ordering of the tasks and the possible assignments of the task variables. The variables can typically represent resources assigned to and/or demanded by the tasks. Further constraints can express conditions on the states of the world before and after executing the tasks. Tasks can be decomposed into subtasks. Planning progresses then through the stages of top-down task decomposition, constraint satisfaction and conflict resolution.

4. PROCESS PLANNING WITH HIERARCHICAL TASK NETWORKS

In the proposed approach both the representation formalism and the planning engine of SHOP, a domain-independent HTN planner were used [11,12]. This planner was provided with domain-specific knowledge concerning parts, features, tools, setups and machines, and extended with search methods aimed at finding the best (i.e., the cheapest) plans and supporting mixed-initiative, interactive planning. All in all, our CAPP system deals with geometry analysis, setup planning, selection and ordering of machining operations and the assignment of the resources.

By applying SHOP we combined the domain-specific and the domain-independent aspects of planning. Domain-specific planner – our CAPP system – is encoded in the domain description language of SHOP. Then the planning engine of SHOP compiles and runs this planner.

4.1. The Model

The planning problem is specified in the problem domain description language of SHOP. This input contains the feature-based model of the part, as well as the enumeration and geometrical models of the applicable tools.

The raw material can either be a prismatic or rotational stock or a pre-product with some features in an advanced state. Its fixturing can be done using a vise or a classical 6-points holding device. The part orientation, the holding device and the positioning surfaces determine together the fixtures.

Technological instructions on when and how to use the tools and fixtures are also part of the domain model. A process plan is a totally ordered sequence of machining and fixturing operations. The cost of a plan is determined by the costs of the individual machining operations plus the costs of resource (tool and fixture) changes.

The task is to generate alternatives of executable and close-to-optimal process plans.

4.2. System Structure

The planning system has several modules. A pre-processor unit analyses the part and generates the operations needed to manufacture the part, as well as the constraints on their ordering. Different modules deal with the selection of appropriate fixtures and tools, and handle the resource assignment and ordering of the operations. Plan correctness is guarded by a body of technological constraints that are expressed as axioms of the CAPP domain. A machine-tool specific module performs simple geometric reasoning. We intend to extend this component to a general-purpose geometric reasoner (see Section 5).

The task of the core, generic planning engine is to solve the planning problem by harmonizing the work of the modules: it has to assign fixtures and tools to the operations and to organize them into an executable, correct sequential plan in the most economical way.

4.3. Hierarchical Planning

Given the feature-based model, the pre-processor composes the list of required operations in compliance with the initial and required states and geometrical parameters of the features. It also recognizes *precedence constraints* between operations and translates strict tolerances between features into statements asserting that manufacturing of those features should

be done in the same setup. Such constraints express *setup coincidence*.

Since the pre-processing determines the operations and their execution costs, which are supposed not to depend on the tool selection, the optimization can be performed by finding an adequate plan and resource assignment with a minimal number of setup and tool changes. According to the hierarchical structure of the manufacturing process, planning is executed at the levels of setup planning, tool selection and operation sequencing.

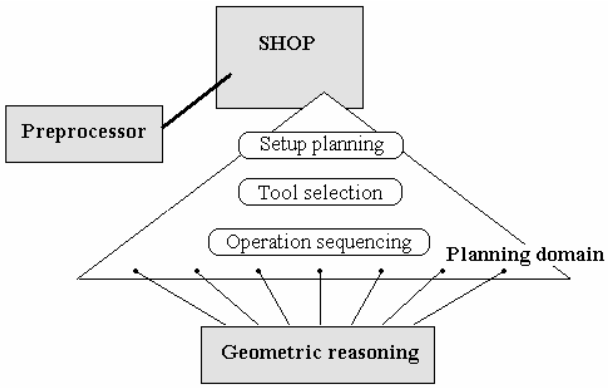

Fig. 1. Architecture of the CAPP system

At the highest, setup planning level the aim of finding a minimal grouping of operations is achieved by an iterative deepening search that minimizes the number of setups. For each group of operations a fixture is selected. While investigating the feasibility of the partial plan, we have to consider the following factors:

- *Precedence constraints*: Groups of operations are subject to constraints inherited from the contained operations;
- *Setup coincidence constraints*;
- *Resource constraints*: There should to be at least one tool available for each operation, able to execute it and being potentially able to access the feature realized by the operation. 'Potentially' means here that the workpiece orientation is convenient, and that the operation can only be obstructed by the removal volume of another feature.

Similarly, the tool selection is also performed by an iterative deepening search according to the number of tools, separately for each setup. Operations that can share tools are collected in sub-groups. However, the decision here is less complicated, since we have only to examine whether there is a tool to execute all operations belonging to a sub-group.

At the operation sequencing level the resource assignment is completed. The operations using the same resources are organized into operation clusters. Hence, only their total ordering is to be established, in accordance with the precedence constraints. There are two types of such constraints. *Explicit precedences* can be fetched from the statement list of the planner (either recognized by the pre-processing module or specified by the user). *Implicit precedences* can only be obtained by an on-line test, assuming full knowledge of the state of the part at the moment of the operation in question. This information is provided by the forward-chaining search mechanism of SHOP. An example of such hard-to-recognize constraints is shown in Fig. 2.

Consider a part with an overlapping hole and a slot, where the hole is deeper than the length of our twist drill. The hole can be drilled only after machining the slot, assuming that the latter is wide enough to admit the chuck. Geometric reasoning is dedicated to investigate such questions and to verify whether an operation can be inserted into the plan as the next step.

This kind of precedence constraint, if realized only during operation sequencing, can make it necessary to backtrack, and in the worst cases, even to reconsider the tooling or the setup plan. However, the experience shows that such situations occur quite rarely and the decomposition achieved by the hierarchical planning offers efficient means to cope with the enormous search space and results in reasonable technological plans.

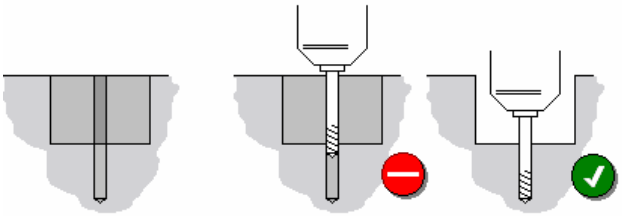

Fig.2. Implicit precedence constraint

5. GEOMETRIC REASONING

Geometric reasoning in CAPP systems has to be accomplished by performing collision tests between solids that represent the part, its features, the machine tool and the fixtures, as well as the tool path. Our planner handles now special cases that are characteristic to the problem instances, but we intend to perform geometric reasoning in a more principled way. Below we describe shortly the basic concepts of our model and algorithms.

Solid models are built of *primitive solids* of simple shapes (box, cylinder). They represent the stock, the removal volume of a feature, or a

component of a tool or fixture.

Compound solids are constructed by applying regularized Boolean set operators to primitive solids. Using De Morgan's laws, the resulting CSG tree can be converted to a special tree, in which the operator *complement* is applied only on primitives. Hence, a compound solid is (1) a primitive solid, (2) the complement of a primitive solid, (3) the union, or (4) the intersection of two compound solids.

We make a reasonable restriction on primitives: they can either be independent or overlapped, but not tangent. Thus, the intersection of any two of them is a finite (maybe empty) set of continuous curves, while that of three primitives is a finite (maybe empty) set of discrete points.

Consider two compound solids A and B, where neither of them contains the other (i.e., $A \not\subset B$, and $B \not\subset A$). They are *colliding*, if there exists a point p such that $p \in A \cap B$. Let us call these points *common points*.

The planner has to determine whether two compound solids are colliding or not. Our algorithm accomplishes it according to the *generate-and-test* paradigm [2]: First, candidate points of interaction are searched for; then each candidate is tested whether it is inside both solids.

If the two solids collide, there also exists an intersection of their boundaries: $p' \in b(A) \cap b(B)$. Furthermore, there is also a common point in the finite set of discrete points that is generated as follows:

- For each pair of primitives (Ta, Tb), such that Ta is a constructing primitive of A, and Tb is that of B, an arbitrary point is chosen on each continuous curve segment of $b(Ta) \cap b(Tb)$.
- For each triplet of primitives (Ta, Tb, Tc), in which there are primitives of both A and B, all the intersection points are generated.

Finally, the so-found points are checked if they are inside the solids A and B. The test is first performed on the ponated or negated primitives of the solids. Then, the results are propagated along the arcs of the CSG tree. The following rules are applied:

Let X and Y be two compound solids, and p the point to be tested. Then

1) $p \in X \cap^* Y \Leftrightarrow p \in X \wedge p \in Y$,
2) $p \in X \cup^* Y \Leftrightarrow p \in X \vee p \in Y$.

Note that the restriction we made on primitives is necessary to justify these apparently trivial rules for regular operators.

The collision test succeeds once a common point is found. The negative result means that the two compound solids do not interact, and there is no geometrical hitch to execute the operation on issue.

6. EXPERIMENTS

The introduced approach was validated on a set of rotational and prismatic test parts. As an example, see our result on a vertical, 3-axis machining center that had to produce a prismatic part with 9 features. Features are referred to by their type and normal vector. This experiment was run on a 233 MHz PC with 32 Mbs of memory. It took 11.25 seconds for our CAPP system to generate the following manufacturing operational plan (see Fig. 4).

Noticeably, the hierarchical decomposition, i.e. the tooling being done separately for setups, results in a surplus tool change. In the second setup, there is no reason not to execute the operations with the endmill first, and those with the sidemill after it. However, on machines with an automatic tool change, it means no extra time, since the tool and setup changes can be done simultaneously, and this handicap is a reasonable price for the achieved saving of the search time.

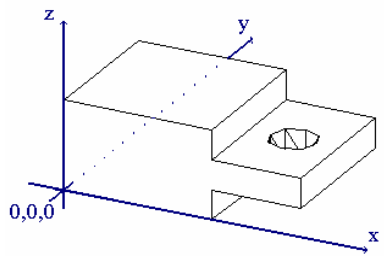

Fig.3. Test part

```
Fit in a vise, z- up, on surfaces x+, x-
        Change tool: 0 ?  sidemill
                Machine face_y-
                Machine face_y+
        Change tool: sidemill ?  endmill
                Machine face_z-
                Rough step_z-
                Finish step_z-
Release
Fit in a vise, z+ up, on surfaces y+, y-
        Change tool: endmill ?  sidemill
                Machine face_x-
                Machine face_x+
        Change tool: sidemill ?  endmill
                Machine face_z+
                Rough step_z+
                Finish step_z+
        Change tool: endmill ?  center-drill
                Center hole
        Change tool: center-drill ?  drill
                Drill hole
Release
```

Fig.4. Manufacturing plan for the test part

7. DISCUSSION AND CONCLUSIONS

In this work we followed the traditional CAPP developments that took the hierarchical approach to planning and optimisation. Such planners first attempted to form optimal setups, and then to sequence the operations optimally relative to this so-called setup plan (e.g., [1,3,14,22]). In contrary to earlier results, we expressed *all* available domain knowledge in the declarative knowledge representation scheme of a general-purpose planner.

The idea of a planning engine that could synthesize the activities and results of several domain-specific reasoning modules appeared in [6]. This process planner worked also with a kind of hierarchical task network. However, neither the HTN engine nor any of the attached reasoning modules concerned with the optimisation criteria. A similar modular architecture appeared in [17] while [21] suggested a blackboard-based platform for integrating the advice of various modules and the decisions of the human planner.

In our previous works we had a two-phased approach to CAPP: In the first phase the search space of solutions were set up by knowledge-based reasoning, and then close-to-optimal solutions were extracted either by genetic algorithms [5] or by case-based reasoning [18]. Recently, we proposed a constraint-based planner engine that could interleave steps of inference, conflict resolution and search [7].

Reasoning and search performed at several levels of abstraction is what makes our hierarchical planner an efficient CAPP system. The first experiments with a limited set of prismatic and rotational parts are encouraging and show the following merits of the proposed approach:

- HTN planning provides a declarative representation scheme with well-defined semantics for capturing all principal kinds of the necessary domain knowledge, including planning strategies, as well.
- The knowledge representation and the plan generation (i.e. search) are separated. Plans can be generated by a sound and complete planning engine. Plan generation can be interleaved with domain-specific (e.g. geometric) reasoning and user interaction which are essential in supporting engineering problem solving.

ACKNOWLEDGEMENT

We are grateful to the creators of the SHOP system for making it freely available.

REFERENCES

[1] Britanik, J., Marefat, M.: Hierarchical Plan Merging with Application to Process Planning. In: *Proc. of the IJCAI'95*, Montreal, 1677—11683, (1995)

[2] Cameron, S.: Efficient Intersection Tests for Objects Defined Constructively. *International Journal of Robotics Research*, **8**(1) 3—25, (1989)

[3] Gupta, S.K., Nau, D.S., Regli, W.C.: IMACS: A Case Study in Real-World Planning. *IEEE Intelligent Systems*, **13**(3), 49—60, (1998)

[4] Halevi, G., Weill, R.D.: *Principles of Process Planning*. Chapman& Hall, 1995.

[5] Horváth, M., Márkus, A., Váncza, J.: Process Planning with Genetic Algorithms on Results of Knowledge-Based Reasoning. *Int. J. of Computer Integrated Manufacturing*, **9**, 145—166, (1996)

[6] Kambhampati, S., Cutkosky, M.R., Tenenbaum, J.M., Lee, S.H.: Integrating General Purpose Planners and Specialized Reasoners: Case Study of a Hybrid Planning Architecture. *IEEE Trans. on Systems, Man, and Cybernetics*, **23**(6), 1503—1518, (1993)

[7] Márkus A., Váncza J.: Process Planning with Conditional and Conflicting Advice. *Annals of the CIRP*, **50**(1), (2001), in print

[8] Marri, H.B., Gunasekaran, A., Grieve, R.J.: Computer-Aided Process Planning: A State of the Art. *Int. J. of Advanced Manufacturing Technology*, **14**, 261—268, (1998)

[9] Nau, D. S., Gupta, S. K., Regli, W. C.: AI Planning Versus Manufacturing-Operation Planning: A Case Study. In *Proc. of the IJCAI-95*, Montreal, 1670—1676, (1995)

[10] Nau, D. S., Smith, S. J. J.,Erol, K.: Control Strategies in HTN Planning: Theory versus Practice. In *Proc. of AAAI-98/IAAI-98*, Madison, WI, AAAI Press, 1127—1133, 1998

[11] Nau, D. S.: Documentation for SHOP 1.6.1 and M-SHOP 1.1.1. http://www.cs.umd.edu/projects/shop/, 2000

[12] Nau, D.S, Y. Cao, A. Lotem, and H. Muñoz-Avila. "SHOP: Simple Hierarchical Ordered Planner." In IJCAI-99, pp. 968-973, 1999.

[13] Russel, S., P. Norvig: *Artificial Intelligence. The Modern Approach*, Prentice Hall, 1995

[14] Sarma, S.E., Wright, P.K.: Algorithms for the Minimization of Setups and Tool Changes in "Simply Fixturable" Components in Milling. *Journal of Manufacturing Systems*, **15**(2), 95—112, (1996)

[15] Schreiber, G, et al.: *Knowledge Engineering and Management. The Common KADS Methodology*, The MIT Press, 1999

[16] Shah, J.J., Mantyla, M.: *Parametric and Feature-Based CAD/CAM*. Wiley, 1995.

[17] Teramoto, K., Onosato, M., Iwata, K.: Coordinative Generation of Machining and Fixturing Plans by a Modularized Problem Solver. *Annals of the CIRP*, **47**(1), 437—440, (1998)

[18] Váncza, J., Horváth, M., Stankóczi, Z.: Robotic Inspection Plan Optimization by Case-Based Reasoning. *Journal of Intelligent Manufacturing*, **9**(2), 181-188, (1998)

[19] Weld, D.S: Recent Advances in AI Planning. *AI Magazine*, **20**(2), 93—123, (1999).

[20] Wilkins, D.E., desJardins, M.: A Call for Knowledge-Based Planning. *AI Magazine*, **22**(1), 99—115, (2001).

[21] Zeir, G. van, Kruth, J.-P., Detand, J.: A Conceptual Framework for Interactive and Blackboard-Based CAPP. *Int. J. of Production Research*, **36**(6), 1453—1473, (1998)

[22] Zhang, H.-C., Lin, E.: A Hybrid-Graph Approach for Automated Setup Planning in CAPP. *Robotics and Computer-Integrated Manufacturing*, **15**, 89—100, (1999)