An Iterative and Interactive Approach for Process Planning

Suresh Batchu, Subbarao Kambhampati, Hirode Kartheek & Jami Shah*
College of Engineering and Applied Sciences
Arizona State University, Tempe, AZ 85287

ASU CSE TR 95-023; September 1995.

Abstract

Process planning involves producing machining plans for manufacturing mechanical parts. It requires satisfying a variety of hard and soft geometric, kinematic, tolerance based and economic constraints. Most existing approaches to automating process planning involve doing a global search for a process plan that is optimal with respect to a pre-specified objective function. Such approaches suffer from two important limitations. First, the search space for process plans is too large to facilitate an efficient systematic search. Second and perhaps more important, the evaluation metrics for process plans are very much context dependent, and it is rarely the case that an accurate optimality metric is available *a priori*. To overcome these limitations, in this paper we propose an *iterative* framework that continuously searches for a better plan based on *interaction* with the user. The search is modeled as hill-climbing with the objective function changing as a result of interaction. We will discuss how the approch can be implemented within ASUFTB, a modern feature-based manufacturing system.

*Batchu (*batchu@asu.edu*) and Kambhampati (*rao@asu.edu*) are with Computer Science department, Hirode (*hirode@enuxsa.eas.asu.edu*) and Shah (*jshah@asuvax.eas.asu.edu*) are with Mechanical Engineering department. This research is supported in part by NSF research initiation award (RIA) IRI-9210997, NSF young investigator award (NYI) IRI-9457634 and ARPA/Rome Laboratory planning initiative grant F30602-93-C-0039 to Kambhampati.

1 Introduction

Process planning involves determining the sequence of operations to perform to manufacture a part given its description and the specification of the resources in the workshop. It should take into account both technological and economic considerations, some of which are hard constraints and some preferences. This knowledge often represents both the experience and know-how of engineers/specialists, which differ from one company to another. Many approaches [13, 8, 3, 14] have been proposed for automating process planning which focus on generating a single plan that is optimal with respect to some predefined criteria. The process of finding a plan involves examining and evaluating the manufacturability of different machining plans. It is complicated by the fact that the search space is huge and the plan evaluation criteria are not obvious at the beginning and can only be gleaned through interaction with the user. Trade-offs among optimization objectives typically reflect user preferences and the presence of additional domain constraints not captured in the planning and scheduling model. Moreover, the user may change the optimality criteria for the process planning during the process of finding an optimal solution. Since the optimality metric is not completely known, we would like to take the human expert/user to be the final arbiter on the quality of the plan produced. Accordingly, if the expert is not satisfied, the planner should be able to resume its search for an improved process plan. To support this, we propose an *iterative and interactive* approach to process planning.

Figure 1 gives the general idea of our proposed approach. The planner starts with a default plan (also called the seed plan) and does hill climbing on its current objective function. Initially, this objective function is set to the default optimality metric. It generates some local repairs based on the current optimality criterion and ranks the effect of each local repair. Those repairs which look promising are picked and are applied to the current plan. Once the planner reaches a minima with respect to its current objective function, the criticism is requested from the user. If the user is satisfied with the current plan he may terminate the process; else he will criticize the plan and this criticism is incorporated into the current plan, this is then made the new seed plan and the process continues. Conceptually, the interaction between the user and the planner can be characterized in terms of a shift in the objective function being used by the planner. This approach is advantageous in that the frequency of interaction is reasonably low and is used only to validate a process plan or change the objective function.

In order to implement this approach, we need to structure the interaction between the planner and the user, and also determine the details of the planner's iterative search process. In particular, we need to answer the following questions: (a) how are plans evaluated and (b) how are they revised on the basis of evaluation. The latter question in turns raises other more detailed questions: (i) when does the planner modify its current seed plan (ii) how does the planner decide which part of the plan to modify? (iii) what is the space of modifications that is allowed? and (iv) how are the modifications carried out? In the rest of this paper, we will address these issues in the context of the ASU Feature Testbed (**ASUFTB**) [4], a feature based

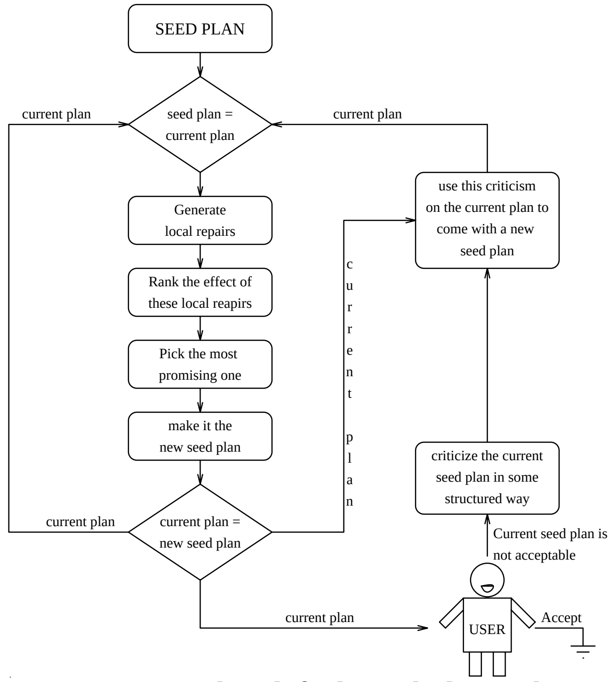


Figure 1: Iterative Planning: Starts with a default seed plan and generate local repairs based on the current optimality criteria. Apply the most promising of these repairs and show it to user. If the user is not satisfied with the current plan he would criticize it and his criticism is incorporated into the current plan. The current plan is then made the new seed plan and it goes through the entire cycle again

manufacturing system developed at Arizona State University.

The rest of this paper is organized as follows. Section 2 explains how the interaction with the user can form the basis for shifting objective functions during hill climbing search, Section 3 talks about preliminaries and terminology of process planning, Section 4 briefly reviews the ASUFTB, and the representation of process plans within it. Section 5 describes how plans can be evaluated through interaction within ASUFTB. Section 6 talks about the types of modifications for a plan. Section 7 describes a realization of the iteractive and interactive process planning architecture within ASUFTB, and Section 8 summarizes our discussion.

2 Shifting objective functions

In this section we explain conceptually how it is possible for the planner to converge to the user's objective function through interaction.

Suppose we have a collection of objects (say *n* objects) of type **O**. Also, let us assume that each of these objects is represented by *m* attributes $\langle t_1, t_2, \ldots, t_m \rangle$. Now suppose that the user wants to choose some objects from this collection and he picks only those whose attributes satisfy some objective function (they are called as *desired objects*).

When the objective function is known, its value for the objects under consideration can be computed and the object which returns the minimum value for the objective function can be picked. But the exact objective function may not be known in some cases. This is true in most of the real life problems and is especially true in the case of process planning. Even then it is possible to pick a desired object with successive iterations consisting of picking some approximate weights for a linear objective function. The objective function in this case can be represented as $O_f = \sum_{i=1}^{m} w_i * t_i$, where t_i is an attribute and w_i is the weight associated with this value. If more importance needs to be given to the attribute t_j, i.e. if its value needs to be decreased, the weight w_j associated with that tuple value t_j is increased. This successive approximation works as follows. The user gives out the approximate weights for a linear objective function and ask the planner to look for objects which are optimal with respect to this criterion. As soon as the planner finds one he examines its attributes. If he finds it satisfying he calls it the desired object, else the planner resumes the iteration process starting from the current best object after changing the weights for the attributes appropriately.

3 Preliminaries & Terminology

This section will describe some of the technical terminology commonly found in the process planning literature and then talks briefly about the issues related to process planning.

A *part* is the final component created by executing a set of machining operations on a piece of *stock* (see Figure 3 for an example). A *stock* is a raw material from which a *part* is machined; it is a shape before any machining is done. A *work piece* is a transient, intermediate state of the stock as it is being machined. Process planning involves finding a set of machining operations to convert a stock into a part.

Objects are represented as a collection of features. A set of features which define the object in its entirety is said to constitute a *Feature Based Model* (FBM). Design features are stereotypical shapes related to a part's function, design intent, or model construction methodology. Manufacturing features are stereotypical shapes that can be made by standard manufacturing operations. Thus, features are application (viewpoint) dependent and so it is possible to represent the same object with more than one FBM.

In a machining operation, material is removed by relative motion between the cutting tool and the workpiece. A machining plan is a sequence of these machining operations which are capable of manufacturing the object from the stock. A *machining feature* is that portion of the workpiece affected by a machining operation. Since an FBM is a collection of application specific features and machining operations are defined in terms of machining features, we need to map these application specific features in FBM onto machining features in order to be able to machine the part. There may be more than one way to perform this mapping, thus there can be more than one machining plan.

4 The ASU features test bed (ASUFTB)

Shah et al. have developed a system called ASUFTB [4] which systematically enumerates alternative features and machining interpretations for an object and these interpretations can

name	*symbol description*
P	Part
S	Stock
R	Removal volume S-P
a	Atomic cell
A	Set of all the atomic cells identified for R
c	Composition of some atomic cells (c $\subset$ A) If two cells need to combine, they should share a common surface. Any number of cells which satisfy this property can be combined into one composite cell. Direction of combination is always perpendicular to this surface
C	Set of all composite cells
m	Machining operation, an instantiation of a template machining operation. All the parameters are known at this instant
f	Feature based model (FBM)
F	Set of all FBMs (f)
ms	Machining sequence, an ordered list of composite cells $\langle c_i, c_j, c_k, \ldots \rangle$
mp	Machining process, an ordered list of machining operations $\langle m_i, m_j, m_k, \ldots \rangle$
wp	Represents the process plan. It is a tuple consisting of the following elements. $\wp$ = $\langle$ A, mapping from A $\rightarrow$ C, mapping/ordering from c $\rightarrow$ f, mapping from f $\rightarrow$ ms $\rangle$

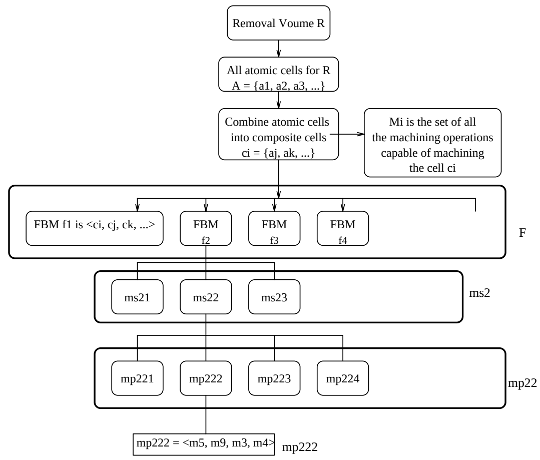


Figure 2: Symbols used for explaining the mathematical representation of ASUFTB. On the right they are represented in a tree form

be used to systematically enumerate all candidate machining plans. Some of the characteristics of ASUFTB which are useful to our discussion are listed below.

1. It generates alternative FBMs. It is possible to generate different machining plans because of this reason.

2. It generates one plan based on default local optimality criteria . We can use it as a base plan and try to modify it so that it is more optimal with respect to some context sensitive criteria.

We will now briefly describe the operation of ASUFTB with the help of the notation in Figure 2 and the example in Figures 3, 4, 5 and 6.

ASUFTB is a design by feature system and uses the following methods to recognize all the machining features. First, the total volume to be removed by machining, called as total removable volume (R) is obtained by subtracting the part (P) from the stock (S). In Figure 3, R is represented by means of dotted lines. This R is decomposed into minimum convex cells, called atomic cells (Figure 3) using a method called halfspace partitioning [4]. A plane cuts the space into two half spaces. Half space can thus be fully characterized with a plane and a direction associated with its normal. All points in space which lie on this side of the plane (i.e. in the direction of this normal) are said to be on the positive side of the halfspace and all the other points are said to be on the negative side of the halfspace. Suppose we need m half spaces $H_1, H_2, \ldots, H_m$ then every atomic cell produced by halfspace partitioning is assigned a m dimensional vector called a *halfspace vector* (HSV). $HSV = \langle d_1, d_2, d_3, \ldots, d_m \rangle$, where component d_1 corresponds to H_1, d_2 to H_2 and so on; d_i is 1 or 0, where 1/0 means the lump lies in the positive/negative half of the corresponding halfspace.

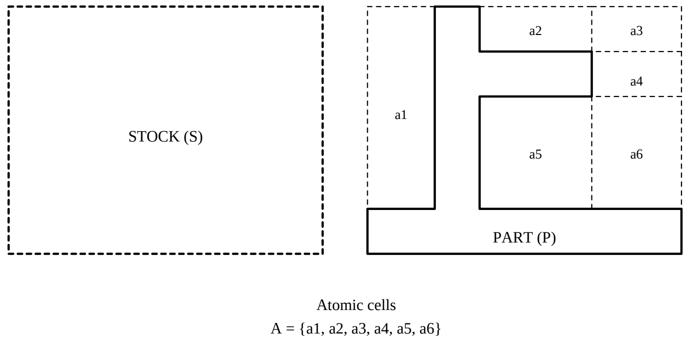


Figure 3: Example: stock(S) and part(P) the removal volume R (S-P) is volume decomposed (using halfspace partitioning) into many atomic cells

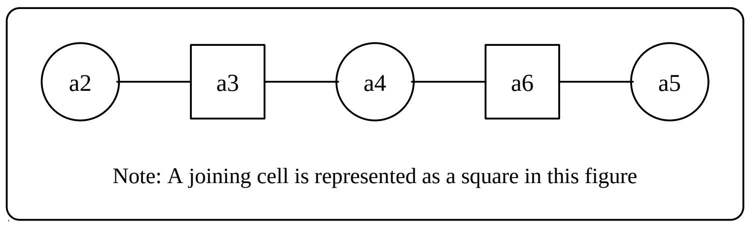
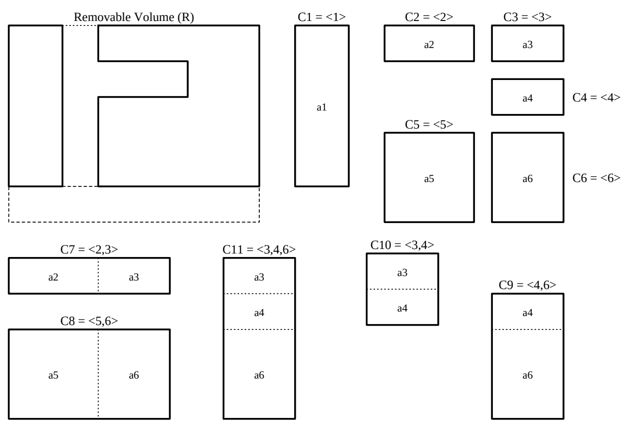


Figure 4: Cell Adjacency Graph (CAG) for the removable volume

The HSVs are used to generate a graph called *cell adjacency graph* (CAG) and an example of this graph for the R in Figure 3 is shown in Figure 4. The nodes in CAG represent atomic cells and arcs represent adjacencies. Two cells are considered to be adjacent if they share a face on the same half space. Note that two atomic cells are adjacent iff they lie in the *same* side of *all* halfspaces *except one*. Some atomic cells, represented as rectangles in Figure 4 need special consideration as these cells serve as crossroads to signify there exist alternative ways of composition. Those special cells are called *joining cells*.

A machining sequence is an ordered list of composite cells by machining which the removable volume R can be removed. Machining sequences are generated from the CAG, as follows. The procedure starts at a joining cell. Adjacent cells are continuously concatenated unless the volume becomes concave. At this stage the concatenated volume is maximally convex and is assumed to be machinable, so it is removed from the CAG and the process begins again at some other joining cell. This process yields alternative trees called *machining sequence trees* (MST). The MSTs generated from Figure 4 are shown in Figure 6. If the original removable volume has n joining cells, the result of the composition will generate n MSTs since this procedure can start at any joining cell. Square nodes in the tree stand for joining cells. Cells in one oval are composed into a composite cell. A path from a root to a leaf defines one machining sequence. FBM is a collection of composite cells where as a machining sequence is an ordered list of composite cells and so there can be more than one machining sequence for a given FBM. A list of FBMs generated for the removable volume R of Figure 3 are shown in Figure 5.



Removal Volume R = {a1, a2, a3, a4, a5, a6}

There are many FBMs possible for this part, but note that not all permutations are possible, the ordering is based on the accessability criteria. Some of the FBMs are listed below.

f1 = {c1, c2, c3, c4, c5, c6}
f2 = {c1, c7, c9, c5}
f3 = {c1, c7, c8, c4}
f4 = {c1, c11, c2, c5}
f5 = {c1, c8, c10, c2}
f6 = {c1, c8, c7, c4}

F = {f1, f2, f3, f4, f5, f6, ...}

Figure 5: Mapping atomic cells into composite cells for the removable volume shown in the previous figures

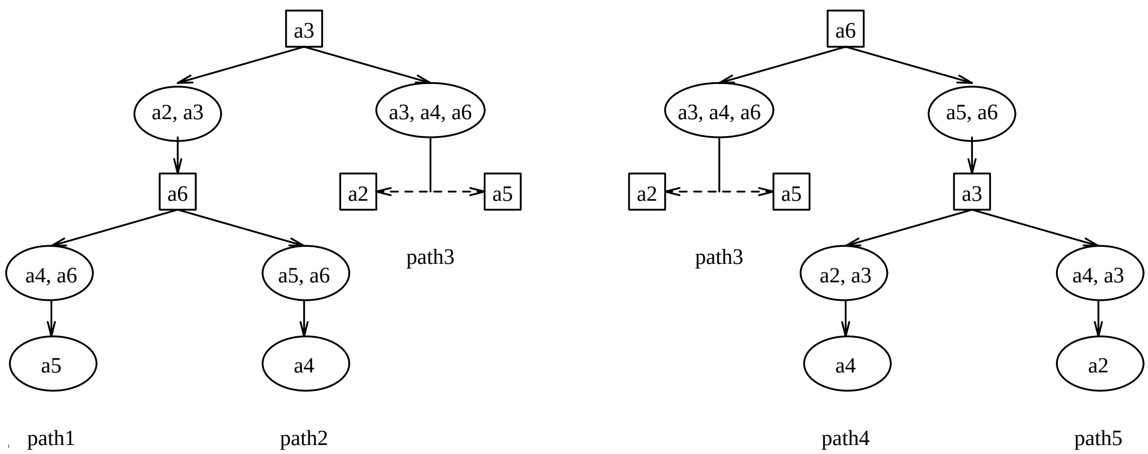


Figure 6: Machining sequence trees (MSTs) obtained from the cell adjacency graph (CAG) shown in the previous figure. Square nodes in the tree stand for joining cells. Cells in one oval are composed into a composite cell. A path from a root to a leaf defines a composition candidate. Composite cells are removed in top down order. Composite cells connected by dashed lines (in the sequence tree) can be machined in any order, as long as their parent is machined first.

<table>
<tr><th rowspan="2">evaluation parameter</th><th colspan="2">parameters affecting the evaluation parameter</th></tr>
<tr><th>product parameter</th><th>process parameter</th></tr>
<tr><td rowspan="2">Feasibility (F)</td><td>part configuration/shape (e.g. intersection of holes, grooves on cylindrical hole, etc.)</td><td rowspan="2">limitations of processes (e.g. drilling for very large holes or grinding of internal, grooved holes</td></tr>
<tr><td>dimensioning and tolerancing (e.g. very long holes, small diameter holes, etc.)</td></tr>
<tr><td rowspan="3">Accuracy (A)</td><td rowspan="3">dimensioning and tolerancing</td><td>process setting (tool related)</td></tr>
<tr><td>fixturing</td></tr>
<tr><td>machine behavior</td></tr>
<tr><td rowspan="2">Consistency (C)</td><td rowspan="2">tolerancing</td><td>process variation (tool related)</td></tr>
<tr><td>machine behavior</td></tr>
<tr><td rowspan="2">Setup time (S)</td><td>the number of different operations required</td><td rowspan="2">versatility of tools</td></tr>
<tr><td>part configuration</td></tr>
<tr><td rowspan="2">Machining time (M)</td><td>dimensioning and tolerancing</td><td rowspan="2">performance of tools (dependent on tool design, tool material, etc.)</td></tr>
<tr><td>the number of different operations required</td></tr>
</table>

Figure 7: Evaluation parameters used to evaluate a process plan. Product parameters and process parameters affect the value of this evaluation parameter

5 Evaluation parameters for a process plan

In order to implement the iterative approach shown in Figure 1 in ASUFTB, we need to know the evaluation parameters for a process plan so as to rank the local repairs. After some preliminary investigation we have settled on feasiblity (F), accuracy (A), consistency (C), setup time (S) and the machining time (M) as the parameters that are important for evaluating the quality of a plan and Figure 7 describes these parameters. The relative weightage given to each of these parameters is context sensitive.

Some of these parameters, such as S and M, can be evaluated automatically by the planner and the other parameters need some context sensitive information (e.g. consistency (C) depends on machine behavior). We have decided to acquire the values for these context sensitive parameters interactively, from a human critic. Each of the above parameters take some values (e.g. *good*, *satisfactory*, *bad*). Finer granularity of these quantizations is also possible. *The aim of iterative planning is to make the resultant plan optimal with respect to each of the* **F, A, C, S, M** *terms by making each of them take one of these two good, satisfactory values*. Thus, at any instant we try to improve the value of that parameter whose current value is *bad*.

Thus the question ‘‘*what parameter to improve?*’’ boils down to picking a parameter which had value bad and changing the plan to improve that parameter.

6 Localizing repair

The next logical questions which need to be addressed are ‘‘*where to modify the plan?*’’ and ‘‘*how to modify the plan?*’’, specifically, what local modifications can be done to the plan to improve the value of the chosen parameter.

6.1 Where to modify the current process plan?

Each machining operation can have the following parameters associated with it and each of these parameters take on some numerical values.

1. Step Setup time **SS**
2. Step Machining time **SM**
3. Step Accuracy **SA**

The overall quality of a plan can be mapped onto these steps (i.e. machining operations) using some sort of greedy approach. For example, if machining time for the entire plan is found to be *bad*, we try to select those machining operations which have high numerical values for the **SM** parameter and try to replace it with some other machining operation which has lower value for this parameter. Let us call all the potential machining operations which need to be replaced in the machining plan the *focal points*. Any heuristic (e.g. greedy approach) can be used to determine the most promising focal points within the problem context.

6.2 How to modify the current process plan?

There are three main ways to modify the plan.

1. Use the same FBM, but replace some of the existing machining operations with different ones. This corresponds to changing the mappings from composite cells to machining operations.
2. Use a different FBM and start looking for better plans for this FBM. As explained in the previous sections, there can be many process plans which are all capable of machining a given FBM (Figure 2) and this technique starts looking at the process plans corresponding to a different FBM with/without fully evaluating all the process plans for the current FBM, and thus corresponds to the commonly used branch and bound technique.
3. Modify the existing FBM so that it results in better plans. This is similar to the previous technique in that it is looking for solutions by changing the FBM, but the way it chooses the new FBM is different. The criticism on the current process plan is used to pick a new FBM. The advantage of this method is that the current mappings/bindings are not lost when the planner moves from one FBM to another FBM in feature space. The planner just disregards all the mappings for features which are no longer present in the new FBM and adds those mappings/features which weren't present in the previous FBM.

The first two of the above three methods are straightforward and so we will elaborate on the third method. Expecting that any arbitrary modification to the existing plan results in a *better* plan is obviously not valid. There are a limited number of distinct methods, called *tweaks*, which can be used to modify the plan and expect the improvement in the desired attribute value. We have identified two of these tweaks and they are discussed below.

1. *Reordering* of features in the FBM. This corresponds to changing the order of sequence of features, thus resulting in the change of FBM. This may or may not result in the reordering of machining operations.

2. *Split and merge* of the composite cells in the FBM. This will result in a different FBM and this method provides an easier way for traversing the search space consisting of the combinations of all the atomic cells. An example of using this tweak on the removable volume R shown in Figure 3 is described below. Please note that this example is very simplistic and many details have been omitted for the sake of clarity, but it provides the gist of the split and merge technique.

   Let the current FBM be $f_5 = \langle c_1, c_8, c_{10}, c_2 \rangle$ (see Figure 5). Let the corresponding machining operations be $\langle cutting, milling, cutting, cutting \rangle$. Let the machining time for the milling operation be large when compared to the other operations. Suppose the critic sees the total machining time and rates it as bad. Using the approach mentioned above for finding the focal points we deduce that we need to improve the machining time for the milling operation. Our system immediately examines the template for the milling operation and finds that the machining time is proportional to the amount of material being removed, now it tries to reduce the amount of material being removed during that milling operation. After trying out some reorderings it tries to do split and merge. The planner starts with f_5 and it tries to split the composite cell c_8 into its components $\{a_5, a_6\}$. This is the splitting stage of this operation. At the end of this stage, we have $\{c_1, a_5, a_6, c_{10}, c_2\}$. The planner then tries to merge the cells which have been split with the adjoining composite cells, in this case it is c_{10}. The atomic cell a_6 can combine with c_{10} since they share a surface; the result of this combination is composite cell c_{11}. Thus, the resulting FBM is $f_{new} = \langle c_1, c_{11}, c_5, c_2 \rangle$ and the corresponding machining sequence is $\langle cutting, cutting, milling, cutting \rangle$. It adds up the machining times and sees that the machining time has reduced considerably (since less material is being removed by the milling operation), it then asks the external critic to evaluate this plan and this cycle is repeated till the external critic is satisfied.

Both these tweaks can affect all the evaluation parameters (F, A, C, S, M); the magnitude of the effect on each parameter will differ from case to case.

7 Iterative architecture on ASUFTB

In this section we will put together the discussion in the previous sections to outline an algorithm (shown in Figure 8) for doing iterative and interactive process planning in ASUFTB. This algorithm starts with a default machining sequence and picks one process plan using some simple heuristic. This process plan is optimal with respect to a given objective function and these functions are represented as linear combinations of some arbitrary weights and the parameter (e.g., M, S, etc.) values. If the user is satisfied with this plan the algorithm terminates, else the criticism from the user is requested. If the user criticizes the plan by changing the weights, a new process plan consistent with the new objective function is generated for the same machining sequence. If the weights are not modified it means that the critic is sure about the objective function, but not with the process plan. So a new machining sequence is found by splitting one composite cell present in the current machining sequence

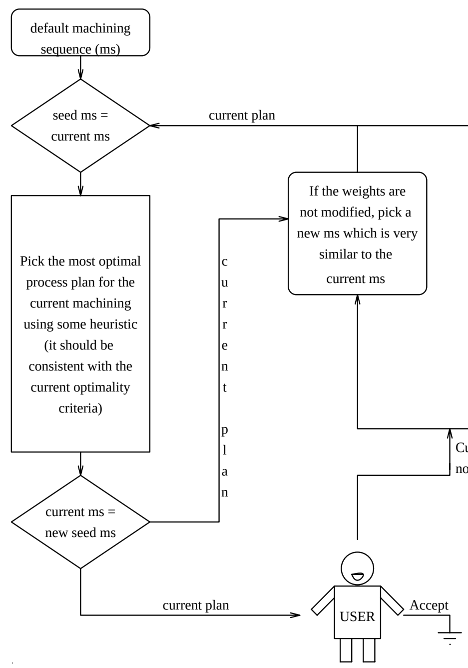


Figure 8: Our interpretation of the Iterative Planning and its usage in our algorithm. We use a heuristic to pick the most optimal process plan for the current machining sequence so that we don‘t need to work on generating local repairs and rank them. Everything else remains same

and a process plan consistent with the same objective function is found. It can be seen that this algorithm is very similar to the one shown in Figure 1.

In the previous sections we discussed the concept of dynamically changing weights to direct the search. Even though the user changed the weights the planner still needs to examine all the objects (or some limited number of them) to pick the next best object, i.e., the next object which is examined has got nothing to do with the characteristics of the current object. Directing this search is possible iff change in weights could be mapped from the current object to the next object. In other words, when the user changes weights, the mapping system should make a rough guess of what transformation is needed and point to an object with the desired attribute values. This technique is extremely useful when the number of objects under consideration is quite large and examining each object in every iteration is expensive. The split and merge technique which we discussed in the earlier section provides this type of mapping and can be used in this algorithm.

The objects under consideration are the machining sequences generated by the half space partitioning. Though there are many attributes of a process plan which can be taken into consideration, we have decided to use machining time (M), setup time (S) and the number of setups (N)[1] as the attributes for each machining sequence. Every machining operation has

[1] In our preliminary discussions with the developers of ASUFTB we have found that their system assumes every machining operation needs a new set up. Thus the number of setups needed for machining one particular machining sequence is equal to the number of composite cells in that sequence.

M and S associated with it. The values for these attributes are generally obtained by using some sort of table lookup on the data given by the manufacturer of the machine; they are also dependent on the characteristics and dimensions of the volume of the material removed in that operation. The machining time of the entire process plan is assumed to be obtained by the addition of the value of M of the individual composite cells; same goes with setup times too. *This means that reordering the features has no effect on the values of the attributes of the process plan when everything else remains same*. This is the reason why we would not be dealing with reorderings any further.

Since every composite cell could be machined in a variety of ways there may be more than one process plan (a sequence of machining operations) for any given machining sequence. We are more interested in providing movement between two machining sequences rather than generating a truly optimal process plan for any given machining sequence and so we have settled for a simple approach of finding an optimal machining operation for a given composite cell given the weights for the corresponding attributes (M, S, N). Since every machining operation has some values for the corresponding attributes, a simple heuristic which picks that machining operation whose attribute values when linearly combined with the weights results in the least value of the objective function, seems sufficient for our purpose. If the user wants some particular composite cell to be always machined by some particular machining operation he could lock that operation on that cell either for just one machining sequence or across all the machining sequences.

For any given set of weights, the corresponding optimal process plan for a given machining sequence is found by picking a sequence of machining operations each of which are optimal with respect to the given objective function (using the above heuristic). If the user is not satisfied with that process plan and wants to persist with the same set of weights then it is time to look at the other machining sequences. The current process plan was not acceptable due to the fact that it had high objective function value and one way of reducing this inoptimality is by removing the most expensive machining operation (with respect to the given weights) present. Since every machining operation in this process plan is already the most optimal with respect to the objective function the only way to get rid of the most expensive optimal machining operation is by examining other machining sequences which do not have this composite cell. Since there might be potentially many machining sequences which do not have this composite cell we need a way to choose one among them in such a way that it is very similar to the current machining sequence at hand. The composite cell which was assumed to be the source of inoptimality is divided into its atomic cells. The list of all the composite cells containing any of these atomic cells is found. The list of all the machining sequences containing any of these composite cells is found and the longest match can be computed on each of the sequences.

Thus the user is able to examine alternate interpretations in the form of different machining sequences and he continually modifies the weights in the objective function till he finds a process plan and machining sequence which he feels are optimal.

8 Summary

We argued that iterative and interactive planning is important in some domains where the optimality criteria/objective function is not known apriori. We proposed an architecture that attempts to converge on a correct objective function through interaction with the user. We modeled this interaction using shifting objective functions, the shift being used as criticism on the current plan. We then explained how this architecture can be implemented in a modern process planning environment like ASUFTB. Specifically, we addressed the issues of how a plan is evaluated, how it is modified and how the modification is focussed. We believe that implementation of this type of process planning approach can provide the right balance between completely automated vs. user-assisted process planning.

References

[1] A.E. Howe, Paul R. Cohen, John R. Dixon and Melvin K. Simmons. Dominic I: Progress Towards Domain Independence in Design by Iterative Redesign. Engineering with Computers, Vol 2, 1987, pp137-145.

[2] Arvind Shirur. Automatic Generation of Machining Alternatives For Machining Volumes. `LD179.15 1994 .S557, ASU`.

[3] Satyandra K. Gupta, Dana S. Nau, William C. Regli and Guangming Zhang. A Methodology For Systematic Generation And Evaluation Of Alternative Operation Plans. Advances in Feature Based Manufacturing, 1994, pp161-184.

[4] Jami J. Shah, Yan Shen and Arvind Shirur. Determination of Machining Volumes from Extensible Sets of Design Features. Advances in Feature Based Manufacturing, 1994. Chapter 7, pp 129-157.

[5] Mark F. Orelup, Paul R. Cohen, John R. Dixon and Melvin K. Simmons. Dominic II: Meta-Level Control in Iterative Redesign. AAAI Proceedings, 1988, pp 25-30.

[6] Matthew L. Ginsberg and David A. McAllester. GSAT and Dynamic Backtracking. KR94

[7] Matthew L. Ginsberg. Dynamic Backtracking. Journal of Artificial Intelligence Research 1 (1993). pp 25-46.

[8] Subbarao Kambhampati, M.R. Cutkosky, J.M.Tenenbaum and S.H. Lee. Integrating General Purpose Planners and Specialized Reasoners: Case Study of a Hybrid Planning Architecture. IEEE transactions on Systems, Man and Cybernetics, Special Issue on Planning, Scheduling and Control, 1993, pp 23(6).

[9] David A. McAllester. Partial Order Backtracking. `http://www.ai.mit.edu/people/dam/dynamic.ps`, 1993.

[10] S.K. Gupta, D.S. Nau, and G.M. Zhang. Estimation of achievable tolerances. TR-93-44, ISR, University of Maryland, College Park, 1993.

[11] N.C. Ide. Integration of process planning and solid modeling through design by features. Master's thesis, University of Maryland, Department of Computer Science, 1987.

[12] Raghu Karinthi and Dana S. Nau. An algebraic approach to feature interactions. *IEEE Trans. Pattern Analysis and Machine Intelligence*

[13] Yannick Descotte and Jean-Claude Latombe. Making Compromises Among Antagonist Constraints in a Planner. Artificial Intelligence, 1985b, 27:183-217.

[14] Yolanda Gil and M. Alicia Perez. Applying a General-Purpose Planning and Learning Architecture To Process Planning. Fall AAAI Symposium series: Symposium on Planning and Learning