Integrating AI and Solid Modeling for Design and Process Planning

Dana S. Nau* Raghu Karinthi† George Vanecek‡
Qiang Yang§

University of Maryland
College Park, MD 20742
USA

Abstract

This paper describes our work on the integration of techniques for solid modeling, geometric reasoning, and multi-goal planning, with application to computer-aided design and manufacturing. This work is being done with two long-term goals in mind: the development of a practical integrated system for designing metal parts and planning their manufacture, and the investigation of fundamental issues in representing and reasoning about three-dimensional objects. We believe this work will have utility not only for automated manufacturing, but also for other problems in design and multi-goal planning.

1. Introduction

One of the greatest problems facing the manufacturing industry today is caused by the different product descriptions used in various segments of the industry. Many tools have been created to aid in the design and the manufacturing processes separately, but major problems remain in developing ways to integrate these tools. One example of this is the integration of automated process planning systems with systems for solid modeling and computer-aided design.

Our first work in this area was directed toward the task of process selection. It was (and still is) our thesis that the rule-based approach used in most knowledge-based systems is not the most appropriate way to do process planning, and that an approach we have developed based on hierarchical abstraction will work better. Based on this idea, we first developed SIPP, a process selection system written in Prolog, and later developed SIPS, a more sophisticated system written in Lisp. The structure of SIPP and SIPS and the continuing development of SIPS over the last

*Computer Science Department, Institute for Advanced Computer Studies, and Systems Research Center. This work was supported in part by an NSF Presidential Young Investigator award, with matching funds provided by Texas Instruments and General Motors Research Laboratories. *E-mail address: nau@mimsy.umd.edu.*

†Department of Computer Science. *E-mail address: raghu@brillig.umd.edu.*

‡Department of Computer Science. *E-mail address: vanecek@tove.umd.edu.*

§Department of Computer Science. *E-mail address: yang@brillig.umd.edu.*

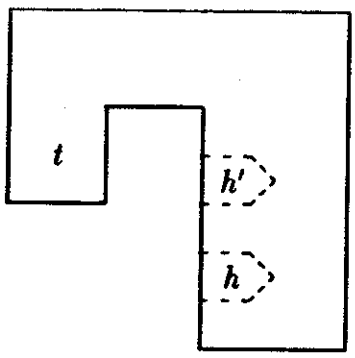

Figure 1: A side view of an object having a tab and two holes.

several years have been described elsewhere [13, 14, 15, 16, 18, 20, 21], so they will not be described again here.

For further work in this area, we have increasingly become interested in design and solid modeling. There are several reasons for this. First, a good design system is essential to provide a decent interface to a process planning system. Second, there are process planning tasks which cannot be performed correctly without extensive interactions with a solid modeler. Our current work concentrates on the integration of design with process planning, through the development of new approaches to solid modeling and geometric reasoning. We are focusing on the following issues:

1. solid modeling techniques specifically suited for integration with automated reasoning systems such as process planning systems;

2. computer-aided design systems capable of reasoning about three-dimensional objects, both for use as a design aid and also for use in integrating design with process planning;

3. ways to reason about interacting features during design and planning.

These issues are discussed in Sections 2-4, respectively. Section 5 contains concluding remarks.

2. Solid Modeling

Most approaches to the integration of solid modeling with automated process planning have essentially involved using a geometric modeler as a front end to a process planning system. Two examples of this involve the use of SIPS as the process planning system: (1) the work done at General Motors Research Labs [21] with MBF/X-Solid, to SIPS, and (2) the work done at the National Bureau of Standards [2] with Unicad/Romulus [24].

This kind of approach is good for making the system more convenient for the designer, but in order to generate correct process plans for complex objects, this approach is not sufficient. Which processes can be used to make a machinable feature—or whether the feature can even be made at all—may depend on geometric information not available solely from the description of the feature.

For example, consider the object whose side view is shown in Figure 1. The hole h can be made by drilling, but there is no way to make the hole h' because the tab

t interferes with the tool trajectory. A process planning system which tries to plan the creation of h' without knowing about the geometry of the rest of object will fail to realize that this is a problem.

In order to produce correct process plans for complex objects, it will be necessary for the process planning system to reason about geometric relationships among the various parts of an object. To carry out such reasoning automatically will require extensive interaction between the process planning system and a solid modeler during process planning. We have built an experimental interface between SIPS and the PADL-2 solid modeler for this purpose [8], but our experience with PADL-2 [19, 31] and several other solid modelers has led us to believe that most existing solid modelers are not completely satisfactory for this task. For this reason, we are developing a new solid modeler, called Protosolid. Several of the considerations leading to the design of Protosolid are discussed below:

1. During process planning, the solid modeler will be required to answer many queries and perform many incremental changes to the solid, all in an efficient manner. In existing solid modeling systems, much work has been done on efficient algorithms for operations such as rendering, but not so much work has been done on providing easy and efficient ways to handle the kinds of machine-generated queries and changes necessary for automated process planning.

   As an example, consider the task of finding out whether or not a particular tool trajectory intersects a workpiece. This requires computing the regularized intersection [23] between the tool trajectory volume and the workpiece, and determining whether this intersection is non-empty. This takes time $O(n^2)$ in BRep (boundary representation) modelers, and $O(n^3)$ in CSG (constructive solid geometry) modelers, typically with a rather large constant factor. Because of this, the task may in some cases require several minutes. To plan the manufacture of a complex object, it may be necessary to answer such queries repeatedly—which could thus require several hours.

   To address this problem, Protosolid incorporates some new algorithms for computing set operations on three-dimensional solids, based on non-regular decomposition of three-dimensional space [28, 29, 30]. This allows Protosolid to perform set operations very quickly; its average-case performance appears to be approximately $O(n \log n)$.

2. If two solids are identical except for a small angle of rotation, many solid modelers fail to do set operations on these solids correctly. Protosolid is successful for rotations as small as 0.1 degree, and most of the time succeeds for even smaller rotations.

3. To extract features from a solid model requires easy access to the data structures and to the functions which manipulate them, and it also requires facilities for attaching attributes and geometric constraints to the nominal solid geometry. In existing solid modeling systems, it is often difficult and time-consuming to access low-level geometric entities such as faces, making it difficult to group them together into features. In addition, the languages (e.g., Fortran) in which these modelers are written often do not provide the flexibility necessary to allow rapid development and testing of the new modeling functions which would be required to develop a better interface.

In contrast, Protosolid is designed to provide a wealth of facilities for creating and manipulating forms and data structures relating to the representation of solids. This, we hope, will make it easy to create systems that link intimately to Protosolid in order to extract features, check tool trajectories, create representations of intermediate workpieces, and reason about the workpiece in other ways.

4. Although most physical solids are bounded by 2-manifolds, the result of a regularized set operation on two solids bounded by 2-manifolds need not necessarily be bounded by a 2-manifold [22]. A general algorithm that implements the set operations on solids must therefore be capable of representing and manipulating solids that are not strictly bounded by 2-manifolds—and this causes problems for BRep modelers that use data structures such as winged-edge representations [1]. Thus, Protosolid has been designed to have no trouble handling solids whose boundaries are not manifolds.

Protosolid is being implemented on a TI/Explorer II, using Common Lisp [26]. Lisp was the programming language of choice for several reasons. Programming solid modelers is to a large extent a play on the use of pointers, lists and symbols—and Lisp is a language designed to do just this. Furthermore, the availability of specialized hardware to execute Lisp programs provides the ability to run Lisp programs very quickly, in a software environment well-suited for rapid prototyping of code. The Explorer architecture has a dedicated 36-bit processor with run-time data-type checking, a high-resolution bit-mapped display, and Ethernet based networking. The software environment includes an excellent editor with advanced features such as interpretation and compilation of code within the editor, incremental compilers, dynamic linking and loading, a flexible display-oriented debugging system and other utilities. We now see Lisp as one of the better languages to use for building solid modelers.

3. Feature-Based Design and Analysis

CAD-generated objects can be defined in terms of the complete geometry of the part. The descriptions contain solids, faces, edges and vertices making up the part. For CAM descriptions of the objects, the geometry and topology are the same, but the meaning associated with this geometric structure is different, and dictates a change in the description. A form feature which the designer may think of as a tab sticking out of a block of metal will be considered by the manufacturing engineer to be the material left after some shoulders have been cut out of a larger block of metal.

There have been various solutions proposed to solve this incompatibility; several of them are discussed below.

1. *Automatic feature extraction* consists of automating (algorithmically?) the task of determining the manufacturing features of a part from existing CAD databases such as IGES files, BReps, etc. The general algorithms developed are of the recognise-feature/extract-feature genre. Prominent among these are the ones by Henderson [6], Kyprianou [11], De Floriani [3], and Srinivasan [25].

   Some of the more significant problems with feature extraction are as follows:

1. Some attributes of a machined part cannot be made without reference to a particular feature (for example, the surface finish, corner radius, and machining tolerances of a pocket). When an object is designed without making reference to these features explicitly, it is unclear how to associate the machining specifications with the proper features.
2. It is difficult to extract a feature which intersects or otherwise interacts with other features, without disturbing those other features. For example, in Henderson's feature extraction system, once a feature volume has been recognized it is subtracted from the overall cavity volume—making it impossible to obtain multiple feature interpretations for the same cavity volume (as was required in Examples 2 and 3 earlier).

2. In *design by features*, the user builds a solid model of an object by specifying directly various form features which translate directly into the relevant manufacturing features. Systems for this purpose have been built for designing injection-molded parts [27], aluminum castings, [12], and machined parts [9, 8, 7, 10].

   In the case of machined parts, one problem with design by features is that it requires a significant change in the way a feature is designed. Traditionally, a designer designs a designs a part for functionality, and a process engineer determines what the manufacturable features are. However, design by features places the designer under the constraints of not merely having to design for functionality, but at the same time specify all of the manufacturable features as part of the geometry—a task which the designer is not normally qualified to do. Another problem—that of alternate feature interpretations—is described in the next paragraph.

3. *Human-supervised feature extraction* overcomes one of the problems of design by features, by allowing the designer to design the part in whatever way is most convenient, and then requiring the process engineer to identify the machinable features of the part. A system for this purpose was built at General Motors Research Laboratories, and another is being built at the National Bureau of Standards [2] using Unicad/Romulus [24].

   Human-supervised feature extraction provides a way for a qualified manufacturing engineer to identify the machinable features–but it still does not handle the problem of alternate feature interperetations. Sometimes a machinable part can be specified as a set of machinable features in more than one way—and unless one is very careful to identify all such possible interpretations, this can lead to significant problems in process planning (as was discussed in Examples 2 and 3).

Consideration of the above approaches makes it evident that no matter what approach is used for designing a part, a major problem to be solved is how to handle feature interactions. Geometric interactions among two or more features may lead to several alternative interpretations as to the identity and dimensions of those features—and these interpretations will dictate restrictions on the order in which the features are to be made.

Hayes's *Machinist* system [4, 5] addresses certain aspects of this problem. For example, in Figure 2a, if the tab t is thin enough that drilling the hole h in t

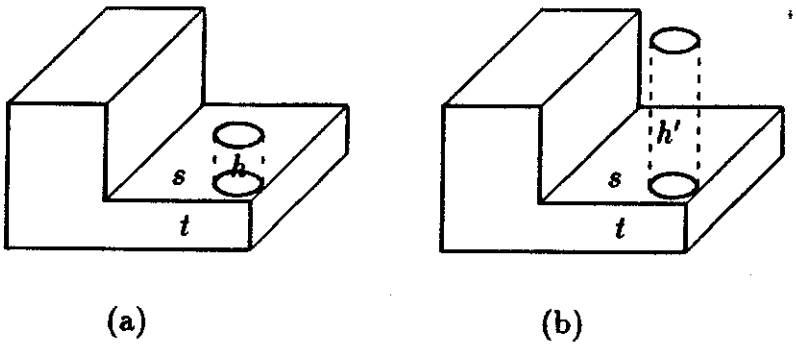

Figure 2: Two different interpretations of a hole.

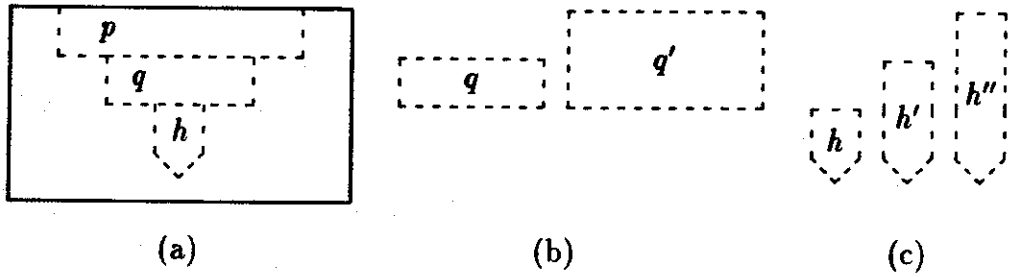

Figure 3: Alternate interpretations of a pocket and a hole in an object.

would cause excessive vibration, Machinist will decide to make h before making the shoulder s. However, since Machinist does not use a solid modeler, it has no notion of geometry. Thus, it does not recognize that if one is making h before s, one is not really making h, but instead the hole h' shown in Figure 2b. Depending on the diameter and depth of h', h' might require a different machining process or cutting tool than h, or it might not even be possible to make h' at all—and Machinist will not recognize this. The output of the Machinist is just an ordering on machining the features.

To address this problem in a more robust manner, we are developing an algebra of feature interactions. This algebra includes a number of operators defined on a set of $2\frac{1}{2}$D subtractive features including holes, pockets, notches, etc. Features are considered to be geometric volumes with associated constraints and attributes. For example, in Figure 3, the hole h may have a number of attributes such as diameter, depth, roundness tolerance, surface finish, etc., and must satisfy the constraint that the top of h opens into the surface defined by the bottom of the pocket q.

The feature algebra will include a number of operations which can be performed on combinations of features to yield other features. For example, in Figure 3, the following operations can be performed:

$$\text{extend}(q, p) = q';$$
$$\text{extend}(h, q) = h';$$
$$\text{extend}(h, q') = \text{extend}(h', p) = h'';$$
$$\text{truncate}(h'', p) = h';$$
$$\text{truncate}(h', q) = \text{truncate}(h'', q') = h.$$

From this, it follows that the object shown in Figure 3a can be produced in several

different ways, by choosing one feature from each of the following sets: $\{p\}$, $\{q, q'\}$, and $\{h, h', h''\}$. Each combination of features from these sets dictates a different sequence in which to make the features. For example, for the combination $\{p, q', h\}$, one would have to make q' before making p, because once p has been made, there is no surface in which to make q'. Furthermore (although it is not the case in this example), some combinations of letters may be incompatible.

Once the features, the operators, and the compatibility and sequencing relationships have been rigorously defined, we plan to implement a feature transformation system. The feature transformation system will be capable of examining the feature descriptions, computing all of the possible interpretations of a set of features, and presenting them to a process planning system such as SIPS.

Consultation with expert machinists has led us to believe that arbitration as to which of the possible feature sets is to be preferred is a matter requiring expertise, which often involves consultation with the process planning system as well as geometric reasoning. For this reason, we intend to integrate the feature transformation system with SIPS, so as to arrive at the best possible choice of features without having to produce complete process plans for every one of them.

Initially, the feature descriptions used as input to the feature transformation system will be created using a simple design-by-features interface to Protosolid. This system will include the ability to specify feature attributes such as tolerances, but will be restricted to work on the set of subtractive features discussed above. However, if time permits, we hope to consider extending this system into a more sophisticated system incorporating the use of "design features," form features which may not correspond directly to manufacturing operations, but which make sense to the designer. This would require the system to translate the design features—and their associated tolerances–into manufacturing features after the design of the part was completed. With an intelligently chosen set of available features and ways for combining them, this should be less complicated than extracting manufacturing features from an ordinary solid model.

4. Reasoning about Interacting Features

The SIPS process selection system works well when the plans for the various features are independent. However, the problem becomes much more complicated when one tries to handle interactions among features (for example, see [4, 5, 32]).

For example, consider an object containing two holes `h1` and `h2`, both having the same diameter and the same machining tolerances. Suppose `h1` can be created by either twist drilling or spade drilling. Then the least costly way to make `h1` is twist drilling. If the depth of `h2` is sufficiently large, `h2` may require spade drilling rather than twist drilling. In this case, the cheapest way to make the entire object is to use spade drilling for both `h1` and `h2` in order to avoid a tool change—even though spade drilling would not be the cheapest way to make `h1` if `h1` were the only hole being made.

The problem described above can be characterized as a problem in multiple-goal planning, with the restriction that all interactions among the actions in the plans should be expressible in terms of partial ordering constraints, identity constraints, and the possibility of "merging" various actions [32]. In the case of process planning, each feature represents a separate goal, and merging corresponds to saving set-up or tool-change costs by performing two operations at the same time (such as the two

twist-drilling actions mentioned above). In such problems, finding an overall plan to achieve all of the goals consists of selecting from among alternate plans for each of the goals and then merging certain of the actions.

As one might expect, the problem of finding an optimal overall plan is NP-hard, but it is possible to develop efficient approximation algorithms for this problem (i.e., algorithms which will produce results that are close to optimal, with reasonably fast average-case performance) [32]. We are developing such algorithms, and intend to develop them further. This will provide a way to produce process plans that take feature interactions into account.

5. Summary and Conclusions

This paper describes our work on the integration of techniques for design, geometric reasoning, and multi-goal planning, with application to computer-aided design and manufacturing. Our work focuses on the following tasks:

1. Knowledge representation and reasoning techniques for process planning. We believe that the rule-based approach normally used in knowledge-based systems is not the best approach to use in process planning. Instead, we have developed an approach based hierarchical abstraction, and implemented it in the SIPS process planning system.

2. Algorithms and data structures for solid modeling. We feel that existing solid modelers are inadequate for the kinds of interactions required for thorough integration with automated process planning systems, and we are addressing this issue by developing a new approach to solid modeling which we believe will satisfy the necessary requirements.

3. Ways to extract and reason about features and feature interactions. We are developing an algebra of feature interactions which will have utility for a number of different approaches to design, including automated feature extraction, design-by-features, and human-supervised feature extraction.

4. Ways to reason about feature interactions and their effects on the resulting plans. We have been developing fast algorithms to handle optimization in multi-goal planning problems, and intend to use these algorithms to handle feature interactions in process planning.

This work is being done with two long-term goals in mind: the development of a practical integrated system for designing metal parts and planning their manufacture, and the investigation of fundamental issues in representing and reasoning about three-dimensional objects. We believe this work will have utility not only for automated manufacturing, but also for other problems in geometric modeling and geometric reasoning.

References

[1] B. Baumgardt, "Winged-edge Polyhedron Representation," Tech. Report CS-320, Computer Science Dept., Stanford University, Stanford, CA, 1972.

[2] P. Brown and S. Ray, "Research Issues in Process Planning at the National Bureau of Standards," *Proc. 19th CIRP International Seminar on Manufacturing Systems*, June 1987, pp. 111-119.

[3] L. De Floriani, "Representation and Exraction of Shape Features in a Solid Model," *NATO ASI on Theoretical Foundations of Computer Graphics and CAD* July 1987.

[4] C. Hayes, "Using Goal Interactions to Guide Planning," *Proceedings of the AAAI-87; the Sixth National Conference on Artificial Intelligence*, 1987, pp. 224-228.

[5] C. Hayes, "Planning in the Machining Domain : Using Goal Interactions to Guide Search", Master's thesis, Carnegie Mellon University, Pittsburgh, 1987.

[6] M. R. Henderson, "Extraction of Feature Information from Three Dimensional CAD Data," Ph.D. Dissertation, Purdue University, May 1984.

[7] K. E. Hummel and S. L. Brooks, "Symbolic Representation of Manufacturing Features for an Automated Process Planning System," Tech. Report BDX-613-3580, Bendix Kansas City Division, 1986.

[8] N. C. Ide, "Integration of Process Planning and Solid Modeling through Design by Features," Master's thesis, University of Maryland, College Park, MD, 1987.

[9] T. Kramer and J. Jun, "The Design Protocol, Part Editor, and Geometry Library on the Vertical Workstation of the Automated Manuacturing Research Facility at the National Bureau of Standards," Internal Report, National Bureau of Standards, Gaithersburg, 1987.

[10] B. Kumar, D. K. Anand, and J. A. Kirk, "Integration and Testing of an Intelligent Feature Extractor within a Flexible Manufacturing Protocol," *Proc. Sixteenth North American Manufacturing Research Conference*, 1988.

[11] L. K. Kyprianou, "Shape Classification in Computer-Aided Design," *Christ's College, University of Cambridge,* 1980.

[12] S. C. Luby, J. R. Dixon, and M. K. Simmons, "Design with Features: Creating and using a Feature Data Base for Evaluation of Manufacturability of Castings," *Computers in Mechanical Engineering* 5:3, 1986, 25-33.

[13] D. S. Nau and T.-C. Chang, "A Knowledge-Based Approach to Generative Process Planning," *Proc. Computer-Aided/Intelligent Process Planning at ASME Winter Annual Meeting*, Miami Beach, FL, November 1985, 65-71.

[14] D. S. Nau and T.-C. Chang, "Hierarchical Representation of Problem-Solving Knowledge in a Frame-Based Process Planning System," *Jour. Intelligent Systems* 1:1, 1986, pp. 29-44.

[15] D. S. Nau and M. Gray, "SIPS: An Application of Hierarchical Knowledge Clustering to Process Planning," *Proc. Symposium on Integrated and Intelligent Manufacturing at ASME Winter Annual Meeting*, Anaheim, CA, Dec. 1986, pp. 219-225.

[16] D. S. Nau and M. Luce, "Knowledge Representation and Reasoning Techniques for Process Planning: Extending SIPS to do Tool Selection," *Proc. 19th CIRP International Seminar on Manufacturing Systems*, June 1987, pp. 91-98.

[17] D. S. Nau "SIPS Command Reference Manual," Tech. Report, Computer Science Department, University of Maryland, 1987.

[18] D. S. Nau and M. Gray, "Hierarchical Knowledge Clustering: A Way to Represent and Use Problem-Solving Knowledge," *in* J. Hendler, ed., *Expert Systems: The User Interface*, Ablex, 1987, pp. 81-98.

[19] E. E. Hartquist and A. Marisa, "PADL-2 Users Manual," Production Automation Project, University of Rochester, 1985.

[20] D. S. Nau, "Hierarchical Abstraction for Process Planning" *Second Internat. Conf. Applications of Artificial Intelligence in Engineering*, 1987.

[21] D. S. Nau, "Automated Process Planning Using Hierarchical Abstraction," *TI Technical Journal*, 1987. Award winner, Texas Instruments 1987 Call for Papers on Industrial Automation.

[22] A. A. G. Requicha, "Mathematical Models of Rigid Solid Objects," Tech. Memo 28, Production Automation Project, University of Rochester, Nov. 1977.

[23] A. A. G. Requicha and H. B. Voelcker, "Boolean Operations in Solid Modeling Boundary Evaluation and Merging Algorithms," *Proc. IEEE* 73:1, 1985, pp. 30-44.

[24] "The Romulus Solid Modeling System, V6.0 User's Reference Manual, Version 1," Shape Data Ltd., 1985.

[25] R. Srinivasan and C. R. Liu, "On Some Important Geometric Issues in Generative Process Planning," *Proc. Symposium on Integrated and Intelligent Manufacturing at ASME Winter Annual Meeting*, Boston, Dec. 1987, pp. 229-244.

[26] G. L. Steele Jr., *Common Lisp: The Language*, Digital Press, Burlington, MA, 1984.

[27] M. Vaghul, J. R. Dixon, G. E. Zinmeister, and M. K. Simmons, "Expert Systems in a CAD Environment: Injection Molding Part Design as an Example," *Proc. 1985 ASME Conference on Computers in Engineering*," 1985.

[28] G. Vanecek, Jr. and D. S. Nau, "Computing Geometric Boolean Operations by Input Directed Decomposition," Computer Science Dept. Tech Report TR-1762, Systems Research Center Tech Report 87-8, 1987.

[29] G. Vanecek, Jr. and Dana S. Nau, "Non-Regular Decomposition: An Efficient Approach for Solving the Polygon Intersection Problem," *Proc. Symposium on Integrated and Intelligent Manufacturing at ASME Winter Annual Meeting*, 1987.

[30] G. Vanecek and D. Nau, "A General Method for Performing Set Operations on Polyhedra," submitted for publication, 1988.

[31] H. B. Voelcker, A. A. G. Requicha, E. E. Hartquist, W. B. Fisher, J. Metzger, R. B. Tilove, N. K. Birrell, W. A. Hunt, G. T. Armstrong, T. F. Check, R. Moote, and J. McSweeney, "The PADL-1.0/2 System for Defining and Displaying Solid Objects," *ACM Computer Graphics* 12:3, Aug. 1978, 257-263.

[32] Q. Yang, D. S. Nau, and J. Hendler, "Optimizing Multiple Goal Plans with Limited Interactions," submitted for publication, 1988.