Enabling Technologies for Automated Redesign

William C. Regli*
National Institute of Standards and Technology
Manufacturing Systems Integration Division
Building 220, Room A-127
Gaithersburg, MD 20899
regli@cme.nist.gov

James Hendler
Computer Science Department
Institute for Advanced Computer Studies
Institute for Systems Research
University of Maryland
College Park, MD 20742 USA
hendler@cs.umd.edu

Dana S. Nau
Computer Science Department
Institute for Advanced Computer Studies
Institute for Systems Research
University of Maryland
College Park, MD 20742 USA
nau@cs.umd.edu

April 28, 1995

Abstract

In this paper we identify AI technologies for enabling interactive automated redesign. We anticipate that these technologies can have great potential impact on future generations of intelligent CAD systems and methodologies.

1 Introduction

Computer-aided design (CAD) and CAD software is fast becoming a ubiquitous component of the modern manufacturing workplace. The decreasing costs of computational power has made sophisticated software for tasks such as finite element and mechanism analysis essential for increasing engineering quality and productivity. Software tools designed to reduce time-consuming build-test-redesign iterations are becoming crucial components for supporting concurrent engineering.

Many of these are tools for design analysis and critiquing. For example, they might examine whether a candidate design violates manufacturing or functional constraints (such as stress, acceleration, and so forth); or they might attempt to find possible suggestions to the user about how to improve a design [17, 14, 22, 5, 31]. Other analysis tools might include those that help the designer foresee potential problems with product life-cycle considerations such as performance, producibility, reliability, maintainability, and so forth.

In order to realize the advantages of collaborative engineering, these design analysis and critiquing systems must consider downstream manufacturing and life-cycle activities during the design phase. This has stretched the limits of traditional design activities and increased their complexity—presenting a variety of difficult computational problems.

*Also affiliated with: Computer Science Department and Institute for Systems Research, University of Maryland, College Park.

The automated redesign problem cuts across all of these issues and is of increasing interest to researchers, in both academia and industry. While some commercial software tools exist (such as those to reduce the number of parts in an assembly), satisfying solutions to the general redesign problem have eluded researchers. Existing systems vary significantly by approach, scope, and level of sophistication, with most attempting to capture manufacturability problems as collections of rules or heuristics. However, it has proven difficult to capture subtle manufacturability problems with hard-coded and coarse rules. Many problems can only be detected at the manufacturing planning level; problems that are compounded when multiple artifacts interact, not only in assemblies, but across the manufacturing enterprise. As a further complication, design is an interactive process and thus all of these computations must be handled in real-time.

This paper is written with several objectives in mind:

- to identify promising new AI technologies for enabling redesign and produce initial outlines for how they may be effectively applied to the real-world manufacturing problem;
- to help overcome two possible risks in the application of AI to computer-aided design: (1) that AI practitioners will apply their technologies to naive or unrealistically simplified versions of the real-world manufacturing problems or (2) that manufacturing engineers will apply the AI technologies in a manner that that does not fully exploit their strengths, ignores their computational costs, or overlooks their representational deficiencies.[1]

We anticipate that this work will serve to further the development of redesign systems by both expanding and improving the application of AI technologies to the problem; leading to the development of systematic methodologies for automated redesign. This will speed the introduction of automated designer's aides that capable of simultaneously considering design goals and manufacturing constraints, and identifying and alleviating manufacturing problems during the design stage.

2 Intelligent Automated Redesign

Many design problems are similar to design problems that have already been solved. Such problems can be approached by taking an existing design and modifying it, rather than producing a new design from scratch. There are several different types of redesign problems:

1. *Redesign for changes to functional specifications.* In many situations, the functional requirements a new design are minor variations on those of a previous design. One approach to solving this problem is to retrieve the old design and adapt it to fit the new requirements. An example of this kind of problem is redesigning a gear box housing to accommodate a larger gear.
2. *Redesign for manufacture with new processes.* The availability of new manufacturing processes introduces the need to redesign products to take advantage of them. For example, engine blocks traditionally were manufactured using casting followed by machining operations. But as die casting becomes a more economical process, the need for lighter cars is leading designers to contemplate the possibility of die-casted engine blocks. Although these engine blocks will have very similar functionality to what they had before, some redesign will be needed to adapt the old designs of engine blocks to the die casting process.
3. *Redesign for changing production resources.* The production resources for an organization change over time: new tools and technologies are added, production resources are prone to failure and downtime, etc. In an agile corporation, meeting the demands of the marketplace might require that products be redesigned to accommodate these changes.

[1] For example, in the early 1980's rule-based expert systems were widely touted as panacea for use in producing solutions to many difficult real-world problems. Although expert systems were successful in some domains and are now in wide use (with several thousand successfully fielded systems), they were also applied to problems for which they were not well suited and produced poor results. The failure of these systems to deliver the results that were promised resulted in much wasted effort.

4. *Redesign for improved manufacturability, reliability, maintainability, etc.* In all component design procedures, the design goes through a design cycle consisting of analysis and review of the design. Now commonly referred to as *design for "X"* (DFX), where "X" can refer to cost effectiveness, quality, or other life cycle considerations such as reliability, maintainability, environmental impact, etc. Ideally, this design phase review should take into account can balance all of the production and performance constraints. This is not possible for all facets of the production process. For example, it is usually only after a component enters the production cycle that experienced process planners and machinists may discover that alterations in the design would be beneficial.

   A current goal is to develop stage tools for design phase analysis that can suggest design revisions, thus helping improve the design's ability to satisfy the constraints imposed by each "X." Our work toward the development of such a tool is described in [3].

This is a problem of increasing interest to researchers, in both academia and industry. For mechanical and electro-mechanical devices, it is much more difficult to reason about the many subtle interactions among the device requirements than it is for purely electrical devices. For example, changing the shape or size of a mechanical housing will change its strength and rigidity in ways that may be hard to predict without doing an extensive analysis (for example, using finite-element techniques).

While some commercial software tools exist (such as those to reduce the number of parts in an assembly), satisfying solutions to the general redesign problem have eluded researchers. Existing systems vary significantly by approach, scope, and level of sophistication. Most automated redesign methodologies employ expert systems and attempt to capture manufacturability problems as collections of rules or heuristics. However, even at the level of individual components, many manufacturability problems are too subtle to be hard-coded in corse rules. The fact that many problems can only be detected at the manufacturing planning level makes it difficult for existing rule-based approaches to capture design difficulties or propose reasonable alternative designs. These problems are compounded when multiple artifacts interact, not only in assemblies, but across the manufacturing enterprise. Further complicating matters is the fact that design is an interactive process and thus all of these computations must be handled in real-time.

An interactive redesign system will need to be capable of analyzing the artifact's design history, its relationship to similar parts in a company's corporate manufacturing databases or files, and the constraints imposed by the different interacting design and manufacturing teams working concurrently on the product. Some of the specific problems to be address are as follows:

- how to represent and reason about partial or incomplete designs;
- how to access and intelligently reuse legacy information (for example, in a corporate knowledge base);
- how to mediate conflicts to satisfy contradictory manufacturing constraints;
- how to provide quick responses for interactive computing environments.

These problems—and some possible approaches for addressing them—are described in the following section.

3 Challenges

3.1 Applications of Plan Retrieval and Reuse

In the area of AI planning systems, a relevant technology is that of *case-based planning*, and particularly the subarea of *plan reuse*. In general, the case based methods focus on the use of a memory of past plans for use in current situations. The analogy in manufacturing is to variant planning approaches. Two aspects of the AI technology may be particularly relevant to manufacturing design – the methods used for the retrieval of past plans and the techniques appropriate for applying the old plans to new situations. Although these are highly related, we treat them here as two separate areas.

3.1.1 Plan Retrieval

Given a set of old plans, there are several techniques that can be used in finding the one (or ones) most appropriate for solving a new problem. The simplest of these techniques is that of feature vectors, representing the plan in terms of a simple string of "keyword" like features. This technique is not dissimilar from the use of group technology codes for variant process planning [2], and thus we will not discuss it further.

More interesting, perhaps, are techniques which work by "indexing" a previous plan based on some set of relevant features arranged in an appropriate data structure for choosing features sequentially with each depending on the previous answer. As an example, a famous program called Chef [20, 18, 19] stored plans for cooking Chinese meals. A sequence of choices were made to decide which previous plan was most like a current one. The first choice might be, for example, to distinguish which type of dish (deep fry, stir fry, bake, etc.). Depending on this choice, the next might be to determine some choice of ingredients (meat or no meat, etc.) Indexed at the leaves of such a "discrimination tree" would be the particular plans for cooking those meals. The advantage of such a scheme is that a large number of plans can be accessed with time logarithmic to the total number of stored plans.

There are several problems with this indexing approach. One is that the set of relevant features must be chosen beforehand. However, if the features are to be of different importance at different times (i.e. sometimes ingredients are important, other times we might care about how long it takes to cook the meal). A second problem is that the features most useful may not be easily identifiable. This means that human intensive "knowledge engineering" work may be required to tie the cases into the indexing scheme. Where this happens, it is difficult to scale this technology to large memories, as would most likely be needed in complex manufacturing domains.

Recent work [1, 13, 25] is focusing on overcoming these problems with indexing by using more efficient, high performance, algorithms to improve memory access. This means that rather than an a priori indexing scheme, patterns of features can be dynamically checked to find relevant plans in memory. This technology allows for the automated creations of case bases and for scaling to the kinds of large memories that will be needed for storing large sets of engineering designs.

3.1.2 Plan Reuse for Manufacturing Planning

Having found a previously used plan, it is necessary to determine how to use it to solve a new problem. In variant process planning systems this is often done by simply displaying an existing process plan and allowing human editing. The techniques of *plan reuse* focus on both automatically identifying those aspects of the existing plan which need to be changed (useful in an interactive system) and in the automated planning of those changed aspects (essentially a combined generative/variant scheme).

The identification of those items needing changing requires two steps. First, a mapping must be identified between the old plan and the new problem. For example, if a previous part had only one drilled hole in it, and the new problem requires two (perhaps with different tolerances or depths), it must be determined which one is the best fit. Although a principled means for doing such mappings efficiently is still an open question, a number of heuristic approaches have been designed.

The second step in identifying (and repairing) changes requires using the mapping, determined in the first step, to direct the refitting of the existing plan for the new problem. Two techniques have shown great promise for this. The first is to develop techniques for abstracting plans into "skeletons" such that a number of specific plans would all have the same high level plan, but with different details. When a mapping is identified, the skeleton that best covers the new problem is chosen. That skeleton is then fleshed out using the details of the current problem. This generates the plan which is expected to solve the problem. One limitation with this approach is that it works best where the skeletons can be automatically identified, and it is unclear what the limitations are on domains that will allow this. [2]

A second approach that shows great promise is that of using "plan annotations" to guide the replanning effort. These annotations are information placed by the planner (human or machine) when it first solves the problem (creating the plan to be stored in memory). Similar to the "design for reuse" framework popular in programming languages, the annotation framework allows information to be stored which keeps track of which items depend on which others, and how various decisions were made. Using this information, efficient approaches have been designed to map and refit existing plans for new problems. To date, this approach has been shown to work with automated (generative) planners in AI domains, and current work is exploring the use of this technique in interactive planning and design systems [23].

[2] To date this technique has been used when the plans are generated in a deductive logic framework, allowing logical inference mechanisms to be used.

3.2 Hierarchical and Partial Information Planning

Engineering design and manufacturing planning each are executed concurrently at several different levels of abstraction. For instance, design proceeds from conceptual level, through embodiment, eventually yielding a detailed design of the product. Similarly, manufacturing planning is done for individual machines, the level of the factory, and enterprise wide. Because it provides a natural way to plan at multiple levels [10, 9, 11], AI techniques for Hierarchical Task-Network (HTN) planning would seem to be ideal for this.

However, some of the barriers to developing the potential of AI planning techniques for planning in practical application domains have been the complexity of HTN planning techniques [8, 7], and the difficulty of integrating them with information about the application domain. AI planners usually represent states of the world as conjuncts of logical atoms (i.e., predicates with arguments), and represent the effects of an operator on the state by adding and deleting atoms from the state. This approach enables AI planners to reason efficiently about partially ordered plans (in which there may be several different possible acceptable orderings for the operators) [24], but it means that such planners cannot easily be used unless the operators' preconditions and effects can easily be represented within the logical formalism.

In domains such as process planning, the preconditions and effects of the planning operators are more naturally represented using solid modeling operations rather than collections of predicates. This can be handled by defining the manufacturing operators as arbitrary pieces of computer code (as in the SIPS process selection system [29] and the Tignum 2 bridge player [32]), or as geometric entities (as in the IMACS system for manufacturability analysis [30, 17, 4]). Such representations make it difficult or impossible to represent partially-instantiated operator preconditions and effects, which makes it very difficult to reason about partially ordered plans—but this difficulty can be circumvented either by generating only totally ordered plans (as in SIPS and Tignum 2), or by first generating totally ordered plans and then removing the ordering constraints after the planner has finished reasoning about the preconditions and effects of the individual operators (as in IMACS).

3.3 Incremental Design and Planning

When performing a planning or design task in many domains it is often difficult to specify in advance what the precise goals are. The process of creating a finished design can be thought of as an *incremental planning* problem, in which an existing plan is incrementally modified to satisfy new or changing goals. The designer specifies goals to the design system, and the design system constructs a design representation that satisfies those goals. To produce the next iteration of the design, the designer specifies new goals, and the design system modifies the existing design to satisfy those new goals.

However, there is one significant difference between this notion of incremental planning and incremental planning as investigated by AI researchers in the past [6, 21, 23]. Since the goals stated by the designer do not necessarily correspond to his/her ultimate intent, in order to produce the next iteration of the design it may be necessary to modify the existing design in ways that violate the goals that led to the existing design. The designer cares less about what particular goals and steps produced the current design than what the current design is, and how it differs from his intentions. It is therefore useful to have a system in which the planning process is performed interactively, with the solution approaching the users intent incrementally through iterations of the planning process. A planning system intended to function in this way must be able to take goal specifications interactively rather than all at once at the beginning of the planning process. The planning process then becomes one of satisfying new goals as they are given by the user, modifying as little as possible the results of previous planning work.

The ability to interactively specify goals enables users to incrementally specify their intent in a design. A planning system that can modify solutions incrementally to match the users' changing intentions allows the system to be used in domains in which it is difficult to specify the goals of the user in advance. For example, [12] describes a system for Civil Engineering design that takes goals from a user interactively and changes the current model to satisfy these goals. The changes to the model are controlled through propagation in a constraint network, thus keeping the model consistent. The system uses a notion of minimal change to insure that the current change affects as few of the users previous intentions as possible. In this way the system allows a designer to incrementally modify a design such that it achieves their intent.

3.4 Search

In general, there may be several alternative ways to manufacture the design. How easy it is to manufacture—or whether it is even possible to manufacture it at all—may depend on which alternative is chosen. Thus, these alternatives should be generated and examined, to determine how well each one balances the need for a quality product against the need for efficient manufacturing.

One difficulty with generating and evaluating alternatives in a brute-force manner is that the number of alternatives can easily grow exponentially with the complexity of the design. One way of preventing the number of alternatives from getting out of hand is to combine branch-and-bound search technique with the use of clever heuristics for pruning unpromising alternatives. This approach has been used to good effect in the generation and evaluation of operation plans [30, 17, 4]). Furthermore, "limited-memory" search procedures such as IDA* [26, 28] and ITS [16] are being developed that can provide optimal or near-optimal solutions very quickly and with only limited memory requirements; and in at least manufacturing problem a limited-memory algorithm has been shown to provide significant improvements over branch-and-bound search [15].

3.5 Accessing and Reusing Legacy Information

As we move toward greater levels of automation in computer aided engineering environments, greater amounts of information can be captured and reused. Information about a design's history and the designers' intent can be recorded during the design process. The design's functional specifications can be modeled and stored in the corporation's databases.

During the design of a new product, tools are needed to give designers efficient and effective access to these masses of data. Further complicating matters is that the integration of this legacy information might require a corporation maintain the legacy data of its business/trading partners. Different parts of major corporations often make commitments to different data formats; likewise, different companies may use different DBMS products (or in-house software) to store their data.

To address problems such as these, it is necessary to develop a methodology for intelligent interchange of diverse, heterogeneous information. A paradigm for integrating multiple heterogeneous databases must be general purpose, i.e. it must be able to provide a "core" set of algorithms that are common across the integration task and are independent of the specific databases being integrated for a given application. This core set of algorithms may then be augmented by application-specific data/subroutines. Systems are being developed (for example, the HERMES system [33, 34, 27]) that run on distribute platforms across the Internet and integrate a wide variety of database and analysis packages. Such systems may also be used for constructing "information-gathering" agents that search the Internet for information that may be of interest in a given application.

4 Discussion

As we move toward greater levels of automation in computer-aided engineering environments, greater amounts of information can be captured and reused. One of the areas with great potential is automated redesign. In this paper we have outlined a number of problem areas to be addressed in the development of automated redesign systems, and have examined the potential use of AI techniques to address these problems.

Although the potential is great, to date this potential is remains largely undeveloped. One reason appears to be the different goals and world views of AI researchers and design researchers, and the mutual lack of familiarity between these two communities. To address this problem, we are beginning the development of a test bed in which to compare AI and manufacturing techniques.[3] We intend to develop a collection of manufacturing design and planning problems and solutions (e.g., designs, plans, and planning systems), presented in a way that is accessible to AI researchers for use as a test set or benchmark set. We hope that this will help AI researchers discover ways to apply AI techniques to manufacturing planning in a realistic manner, and possibly to discover issues arising in manufacturing that may be useful for AI in general.

References

[1] William A. Andersen, James A. Hendler, Matthew P. Evett, and Brian P. Kettler. Massively parallel matching of knowledge structures. In Hiroaki Kitano and James Hendler, editors, *Massively Parallel Artificial Intelligence*, page in press. AAAI Press/The MIT Press, Menlo Park, California, 1994.

[2] Tien-Chien Chang. *Expert Process Planning for Manufacturing*. Addison-Wesley Publishing Co., 1990.

[3] D. Das, S. K. Gupta, and D. Nau. Reducing setup cost by automated generation of redesign suggestions. In Kosuke Ishii, editor, *Proc. ASME Computers in Engineering Conference*, pages 159–170, 1994. Best-paper award winner.

[4] D. Das, S. K. Gupta, and D. Nau. Generating redesign suggestions to reduce setup cost: A step towards automated redesign. *Computer Aided Design*, 1995. To appear.

[5] R. Dighe, M. J. Jakiela, and D. R. Wallace. Structural synthesis under manufacturability constraints: A CAD system for the design of injection-molded product housings. *Research in Engineering Design*, 5(3,4):185–201, 1993.

[6] C. Elkan. Incremental, approximate planning. In *Proceedings of the Eighth AAAI*, pages 145–150, Boston, MA 1990.

[7] K. Erol, J. Hendler, and D. Nau. Complexity results for hierarchical task-network planning. *Annals of Mathematics and Artificial Intelligence*, 1995. To appear.

[8] K. Erol, J. Hendler, and D. S. Nau. HTN planning: Complexity and expressivity. In *AAAI-94*, 1994.

[9] K. Erol, J. Hendler, and D. S. Nau. Semantics for hierarchical task-network planning. Technical Report CS TR-3239, UMIACS TR-94-31, ISR-TR-95-9, University of Maryland, March 1994.

[10] K. Erol, J. Hendler, and D. S. Nau. UMCP: A sound and complete procedure for hierarchical task-network planning. In *Proc. Second International Conf. on AI Planning Systems (AIPS-94)*, pages 249–254, June 1994.

[3] We are expecting ARPA funding for this project in the near future. The PI's are Jim Hendler (hendler@cs.umd.edu) and Dana Nau (nau@cs.umd.edu). The work will be carried out jointly with Steve Ray (ray@cme.nist.gov) at NIST. We solicit your input!

[11] K. Erol, J. Hendler, D. S. Nau, and R. Tsuneto. A critical look at critics in HTN planning. In *IJCAI-95*, 1995. to appear.

[12] D. P. Eshner, J. Hendler, and D. S. Nau. Incremental planning using conceptual graphs. *Journal of Experimental and Theoretical AI*, 4:85–94, 1992.

[13] Matthew P. Evett, James A. Hendler, and William A. Andersen. Massively parallel support for computationally effective recognition queries. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 297–302, Menlo Park, California, 1993. AAAI Press.

[14] R. Gadh, E.L. Gursoz, M.A. Hall, F.B. Prinz, and A.M. Sudhalkar. Feature abstraction in a knowledge-based critique of design. *Manufacturing Review*, 4(2):115–125, 1991.

[15] S. Ghosh, A. Mahanti, R. Nagi, and D. Nau. Manufacturing cell formation by state-space search. 1994. Submitted for publication. ISR TR 93-75.

[16] S. Ghosh, A. Mahanti, and D. S. Nau. Its: An efficient limited-memory heuristic tree search algorithm. In *AAAI-94*, 1994.

[17] S. K. Gupta and D. S. Nau. A systematic approach for analyzing the manufacturability of machined parts. *Computer Aided Design*, 27(5), 1995. To appear.

[18] Kristian Hammond. Chef: a model of case-based planning. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, Philadelphia, Pennsylvania, 1986.

[19] Kristian J. Hammond. Case-based planning: an integrated theory of planning, learning, and memory. Technical Report YALEU/CSD/RR 488, Yale University Department of Computer Science, 1986. (PhD Thesis).

[20] Kristian J. Hammond. *Case-based planning: viewing planning as a memory task*. Harcourt Brace Jovanovich, Boston, Massachusetts, 1989.

[21] B. Hayes-Roth, F. Hayes-Roth, S. Rosenschein, and S. Cammarata. Modeling planning as an incremental, opportunistic process. In *Proceedings IJCAI-79*, pages 375–383, 1979.

[22] M. Jakiela and P. Papalambros. Design and implementation of a prototype intelligent CAD system. *ASME Journal of Mechanisms, Transmission, and Automation in Design*, 111(2), June 1989.

[23] S. Kambhampati and J. Hendler. A validation structure based theory of plan modification and reuse. *Artificial Intelligence*, 1992.

[24] S. Kambhampati and D. S. Nau. On the nature and role of modal truth criteria in planning. *Artificial Intelligence*, 1994. To appear.

[25] Brian P. Kettler, James A. Hendler, William A. Andersen, and Matthew P. Evett. Massively parallel support for case-based planning. *IEEE Expert*, page in press, February 1994.

[26] Richard Korf. Planning as search: A quantitative approach. *Artificial Intelligence*, 33(1):65–88, September 1987.

[27] J. Lu, A. Nerode, and V.S. Subrahmanian. Hybrid knowledge bases. *IEEE Transactions on Knowledge and Data Engineering*, Accepted 1994.

[28] A. Mahanti, D. S. Nau, S. Ghosh, A. K. Pal, and L. N. Kanal. Performance of IDA* on trees and graphs. In *Proc. AAAI-92*, pages 539–544, July 1992.

[29] D. S. Nau. Automated process planning using hierarchical abstraction. *TI Technical Journal*, pages 39–46, Winter 1987. Award winner, Texas Instruments 1987 Call for Papers on AI for Industrial Automation.

[30] D. S. Nau, W. C. Regli, and S. K. Gupta. Ai planning versus manufacturing-operation planning: A case study. 1995.

[31] B.G. Silverman and T.M Mezher. Expert critics in engineering design: Lessons learned and research needs. *AI magazine*, 13(1):45–62, 1992.

[32] S. J. J. Smith and D. S. Nau. A planning approach to declarer play in bridge. *Computational Intelligence*, 1995. Accepted subject to revisions; revisions are in progress.

[33] V.S. Subrahmanian. Amalgamating knowledge bases. *ACM Transactions on Database Systems*, 19(2):291–331, 1994.

[34] V.S. Subrahmanian, S. Adali, A. Brink, R. Emery, J.J. Lu, A. Rajput, T.J. Rogers, R. Ross, and C. Ward. Hermes: A heterogeneous reasoning and mediaton system. *Submitted for publication*, 1995.