Applications

Prospects for Process Selection Using Artificial Intelligence

Dana S. Nau
Computer Science Dept., University of Maryland, College Park, MD 20742, USA

and

Tien-Chien Chang
School of Industrial Engr., Purdue University, West Lafayette, IN 47907, USA

One problem facing modern industry is the lack of a skilled labor force to produce machined parts as has been done in the past. In the near future, this problem may become acute for a number of manufacturing tasks. One such task is process planning. Since process planning requires intelligent reasoning and considerable experiential knowledge, almost all existing computer aided process planning systems require a significant amount of supervision by an experienced human being.

There is some prospect that "expert computer system" techniques from the field of Artificial Intelligence may be successfully used to automate (at least partially) several of the reasoning activities involved with process planning. This paper discusses some current prospects for automating a process planning task known as process selection. These ideas are currently being considered for use in the Automated Manufacturing Research Facility project at the U.S. National Bureau of Standards, and steps are being taken to implement them in an expert computer system.

Keywords: Automated manufacturing, artificial intelligence, CAD, CAM, computer aided design, computer aided manufacturing, expert computer systems, knowledge-based computing, process planning.

North-Holland
Computers in Industry 4 (1983) 253–263

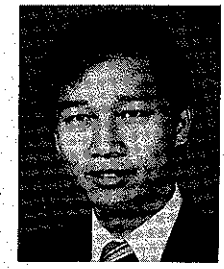
1. Introduction

One major problem facing modern industry is the lack of a skilled labor force to produce machined parts as has been done in the past. This problem has been alleviated to some extent by the development of NC machine tools and APT-like parts programming languages. However, progress at automating some of the higher-level functions involved with manufacturing has not been as fast.

Dana S. Nau is an assistant professor of computer science at the University of Maryland. He received a B.S. in applied mathematics from the University of Missouri in 1974, and received an A.M. and Ph.D. in computer science from Duke University in 1976 and 1979, respectively, where he attended on NSF and James B. Duke graduate fellowships.

Dr. Nau has published more than twenty papers, mainly on subjects related to artificial intelligence. His current research interests include searching and problem-solving methods in artificial intelligence, and expert computer systems.

Dr. Nau's previous research experience includes summer appointments at IBM Research in New York and at the National Bureau of Standards in Maryland. During the latter appointment, he made recommendations for the use of expert computer system techniques in automated manufacturing. He currently has a research contract from the National Bureau of Standards to continue this work.



Dr. Tien-Chien Chang is an Assistant Professor of Industrial Engineering at Purdue University. He obtained his BIE degree from Chung Yuan University, Taiwan, in 1976, and MS and PhD degrees in Industrial Engineering and Operations Research from Virginia Tech, in 1980 and 1982 respectively.

Dr. Chang's research interest is in computer-aided manufacturing. He is co-authoring a book with Dr. Richard A. Wysk, titled *An Introduction to Automated Process Planning* which is to be published by Prentice-Hall in 1984.

Dr. Chang is a member of IIE, SME and Alpha Pi Mu.

Recently, the field of Artificial Intelligence (AI) has advanced to the point that AI projects are accomplishing useful practical results in science and industry. Most of these results involve the design and utilization of *expert systems*, computer systems which use problem-specific problem solving knowledge to achieve a high level of performance in a field which we would normally think of as requiring a human expert. There is some prospect that expert system techniques may be successfully used to automate (at least partially) several of the reasoning activities involved required in the manufacturing process.

1.1. The AMRF Project

One project in which considerable use of expert computer system techniques is planned is the Automated Manufacturing Research Facility (AMRF) project at the National Bureau of Standards.

The AMRF project is an ambitious project whose goal is to build an automated machine shop for manufacturing metal parts. The machining processes currently of interest in the project are those performed by machine tools (lathes, mills, etc.) which do metal removal operations. As envisioned, the machine shop will be automated to as great a degree as the state of the art permits.

The organization of the AMRF is hierarchical in nature [1]. The levels of the hierarchy correspond to the successive division of tasks into smaller and smaller subtasks which are then performed by modules located at various levels of the hierarchy. As illustrated in Fig. 1, a module M accomplishes its assigned task T by dividing T into a series of subtasks $T_1, T_2, \ldots, T_n$, and issuing commands to modules at the next lower level to carry out these subtasks. If a subtask T_i cannot be carried out successfully, then the module responsible for T_i reports failure to module M. In this case, M either selects an alternate series of subtasks T_1', $T_2', \ldots, T_n'$ to accomplish T or, if no alternate series of tasks can be used, reports failure to the level of the hierarchy directly above M. This technique is known in artificial intelligence as *problem reduction*.

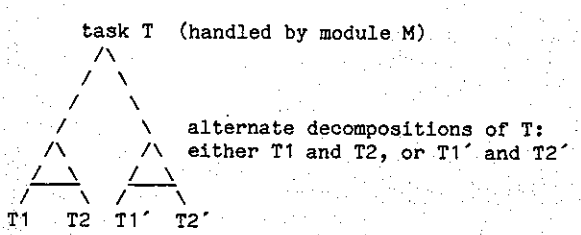


Fig. 1. Problem Reduction on a Task T.

It is planned that considerable use will be made of distributed computing in the AMRF. For example, the control of a single machine tool (which corresponds to a relatively low level in the hierarchy) will probably be done by a small computer which is responsible only for that machine tool, and which receives its commands from another computer responsible for higher levels in the hierarchy.

1.2. Intelligent Reasoning About Manufacturing Tasks

Many of the tasks to be performed automatically in the AMRF require intelligent reasoning. Automating such tasks will require using sophisticated computer programs to perform the experiential reasoning normally done by skilled designers, workmen, and planners. For example, one essential aspect of the manufacture of metal parts in a machine shop is *process planning*, the task of planning the various processes or operations to be made on a metal workpiece to transform it into the desired part. Process planning is a very complex task which requires considerable experiential knowledge, and thus most existing computer aided process planning systems require a significant amount of supervision by an experienced human process planner.

One process planning activity which appears possible to automate at least partially using expert computer system techniques is *process selection*. This is the task of selecting appropriate processes to produce each of the machined surfaces of a part, and providing constraints on the possible sequences in which these processes may be performed.

This paper discusses how expert computer system techniques might be used in the automation of process selection. The techniques discussed in this paper are currently being considered for use in the AMRF, and further research along this line is planned, and an implementation effort has begun [24].

2. Background

This section provides a brief introduction to two different subject areas: expert computer system techniques (Section 2.1) and process selection (Section 2.2).

2.1. Expert System Techniques

Expert systems are computer systems which use problem-specific problem solving knowledge to achieve a high level of performance in a field which we would normally think of as requiring a human expert. It may not be clear from this definition what distinguishes such a system from an ordinary applications program. Certainly, applications programs make use of specialized problem-solving knowledge, and many of them reach high levels of performance. Probably the main difference is that in most expert systems, the model of problem-solving in the application domain is explicitly in view as a separate entity or "knowledge base" rather than appearing only implicitly as part of the coding of the program. This knowledge base is manipulated by a separate, clearly identifiable control strategy. Such a system architecture provides a convenient way to construct sophisticated problem-solving tools for many different domains.

Expert system techniques are now finding their first applications outside of artificial intelligence research laboratories. DENDRAL [14] has been used by university and industrial chemists on a number of problems. R1 [19] is being used by Digital Equipment Corporation to aid in installing computer systems. And Schlumberger's Dipmeter Advisor [9] is currently being adapted for field use after several years of development in the laboratory.

Ordinary computer programs organize knowledge on two levels: data and program. Most expert computer systems, however, organize knowledge on three levels, as follows.

The first (and lowest) level is the data level. This level contains the declarative knowledge about the particular problem being solved, and about the current state of affairs in the attempt to solve the problem. This is analogous to data in any ordinary computer program. The second level contains what is commonly called the *knowledge base*. This is problem solving knowledge which is specific to the particular kind of problem which the system is set up to solve, and which is used by the system in reasoning about the problem. The third (and highest) level contains the control structure. This is a computer program which makes decisions about how to use the problem-specific problem solving knowledge found in the knowledge base. It makes decisions, about how to use the knowledge found in the knowledge base in order to manipulate the data. Computer programs organized in this manner are commonly called *knowledge-based systems.*

The knowledge found in the knowledge base is usually procedural in nature, in the sense that it tells how the data for a problem may be manipulated in order to go about solving the problem. Such knowledge could obviously be represented as a conventional computer program. This is undoubtedly the best way to represent procedural knowledge when the procedure is well-understood, and in fact some expert computer systems have been developed using conventional programming procedures (e.g., [2]). However, artificial intelligence problems are typically ones for which the precise series of steps necessary to solve the problem may not be known, and for which it is necessary to search through a space containing many alternate paths, some of which lead to solutions and some of which do not.

In such situations, it is often useful to encode domain-dependent knowledge in the form of *operators* or *pattern-invoked programs*. These are programs which are not called by other programs in the ordinary way, but instead are activated (by the control structure) whenever certain conditions hold in the data.

A pattern-invoked program can range in size from a single statement (as in MYCIN [10]) to several hundred lines of coding (some of the "knowledge sources" used in Hearsay-II [13] are this large).

It is the responsibility of the control structure to invoke the various operators in the knowledge base as they become applicable to the data. Several operators may be applicable at any given time, in which case the control structure must decide which of the applicable operators to employ.

Several different programming languages, such as PROLOG [12] and OPS [15] have been written which allow for pattern-driven invocation of programs in one form or another. These languages often include features such as automatic back-

tracking if an attempted solution fails, and ways to schedule the pattern-invoked programs if several of them are applicable at the same time.

One type of pattern-invoked program which is of particular interest is the *production rule*, a degenerate program of the form

IF condition THEN primitive action.

The condition is usually a conjunction of predicates which test properties about the current state, and the primitive action is some simple action which changes the current state. For example, the operators given in the 15-puzzle example are production rules. A problem solving system in which the domain-dependent procedural knowledge is represented using production rules is known as a *production system*. Examples of production systems include MYCIN [10], an expert system for medical diagnosis, and R1 [19], a knowledge-based computer system which is currently being used by Digital Equipment Corporation to configure VAX computer systems.

2.2. Process Selection

In several existing computer-aided process planning systems, process selection works as follows. A Group Technology (GT) code [7] is used to classify a part as being in a family of similar parts. When a process plan for a part is desired, a human user enters the GT code for the part into the system, and the system retrieves a process plan which was previously used for some part in this family. The process plan is then modified by the user to produce a plan for the part. Examples of such systems are CAPP [17] and MIAPP [16].

Such systems are quite useful in industry, as they allow process plans to be made very quickly. However, if process planning is to be automated more fully, it will be necessary for the computer system to have more complete information about the part than just a GT code. In particular, detailed information must be available about each surface to be machined.

A few process planning systems have been developed experimentally which use as input a representation of each of the machinable surfaces of the part. By a *machinable surface* we mean any geometric surface or combination of geometric surfaces which can be produced by a single machine tool operation. For example, a hole is a single machined surface, although it consists of both a cylinder and a cone. For more information on machining operations, see [3], Chapter 1.

CPPP [18, 11] is a system which does process selection and sequencing for rotational (lathe-turned) parts. To use CPPP, a user gives CPPP the name of a "part family" containing the desired part, and for each surface to be machined, gives CPPP a code describing the surface.

The part family name is used to retrieve a process model, which is a simple computer program written in a language designed for that purpose. The process model must have been previously written and debugged by a human process planner, and must consider all the possible part features in the family. Because of the restrictions of the process planning language, the process model amounts to a decision tree in which information about the surfaces is used to select and sequence processes.

APPAS [23] is a system which does process selection for both rotational and prismoidal parts. To use APPAS, the user enters for each surface a code describing the surface. The system then uses knowledge of the capabilities of various processes to select a process for each surface. Since APPAS is restricted to rotational and prismoidal components, it need not consider the geometrical relationships which may occur among the surfaces. Thus APPAS selects processes for each surface independently.

CADCAM [5, 8] is an extension of APPAS. The main features of this extension are a graphics interface which allows the user to design a part interactively, and an encoding of the process planning decision logic into decision tables.

3. Automated Process Selection Using AI Techniques

Chang [6] has reviewed 28 process planning systems and found none of them to be adequate for the AMRF. Nau [20] has discussed the various activities necessary for process planning in the AMRF and has found that many of them cannot be automated without further research.

There are a number of factors influencing process selection and sequencing. At present, it is not obvious what all of these factors are, or what effects they might have. It is likely that some of

Table 1. A Frame for a Hole.

Frame type: hole
name: ——— (a unique identifier for the hole, to be filled in when the frame is copied)

slot name	*possible values*	*comments*
bottom	conical or flat	default is conical
length	a real number	length of the hole
pos-length-tol	a real number	pos. tolerance for length
neg-length-tol	a real number	neg. tolerance for length
diameter	a real number	diameter of the hole
pos-diam-tol	a real number	pos. tolerance for diam.
neg-diam-tol	a real number	neg. tolerance for diam.
on-surface	a flat surface name	the surface on which the hole is located
x-loc	a real number	x coordinate of location
y-loc	a real number	y coordinate of location
z-loc	a real number	z coordinate of location
loc-tol	a real number	max acceptable distance from location
x-angle	a real number	angle of tilt to x axis
z-angle	a real number	angle of tilt to z axis
surface-finish	a real number	max acceptable roughness
straightness	a real number	max acceptable deviation
roundness	a real number	max acceptable deviation
parallelism	a real number	max acceptable deviation

the requisite decision-making knowledge can only be gained from experience, as was the case with expert medical diagnosis systems such as MYCIN [10] or CASNET [22]. During the development of a process selection system, it may be necessary to make extensive additions and modifications to the decision logic.

For the reasons given above, it appears that automating the task of process planning could best be done using AI expert system techniques. These techniques could be used to represent knowledge about what information to request from the user, when to request it, and how to use it for process selection.

Table 2
A Frame for a Chamfer.

Frame type: chamfer
name: ——— (a unique identifier for the chamfer, to be filled in when the frame is copied)

slot name	*possible values*	*comments*
hole	a name of a hole	the hole on which the chamfer appears
type	simple linear, inner fillet, outer fillet	different types of chamfers
radius	real	

3.1. Representation of Knowledge About the Part

It is obvious that the AMRF process selection system must make use of descriptions of the surfaces of a part, as do several of the systems described in Section 2.2. However, the AMRF system must be capable of doing process planning for a wider class of parts than just rotational or prismoidal parts. This will require information about the relationships among the surfaces, as well as descriptions of the surfaces themselves.

One way to organize the necessary information about each surface of a part is by using what is known in Artificial Intelligence as a *frame*. Frames are data structures in which all the knowledge about a particular object or event is stored together. Such an organization of knowledge is useful for modularity and accessibility of the knowledge. In addition, frame systems often allow ways to specify default values for pieces of information about an object what that information is not explicitly given.

In the case of a computerized process planning system, for each type of machinable surface in our hypothetical system (as well as for certain other geometric entities), there could be an uninstantiated (i.e., empty) frame representing the possible characteristics of that type of surface. Whenever a user specified a new surface to be machined, the frame corresponding to that type of surface would be copied, and the copy would be used to represent the particular surface specified by the user. During the interaction between the user and the system, the slots of this frame would be filled in with values describing the surface.

Table 1 and Table 2 are possible frames for two types of machinable surfaces: holes and chamfers.

There are several possible ways a computerized process planning system could get the information it needs about each surface. One would be by examining a detailed representation of the part created using a separate computer aided design system. One difficulty with this is that until a working computerized process planning system exists, it is not certain what kind of representation the computer aided design system should create in order to be adequate for the process planning task.

One way to avoid this problem is to have the process planning system request its information directly from a human user rather than using a computer aided design system as an intermediary. Such information could include descriptions of the machinable surfaces, their orientation relative to other surfaces, what tolerances are required, which surfaces are adjacent, what special features are present, and so forth.

Not all of the above information will be necessary for every surface of a part, and it may not be clear at the outset what information will be required for each surface. If the user is required to enter a complete description of the part before the process selection system starts operating, it may turn out that some information is unnecessary and other information is missing. Therefore, it would be better to have an interactive system which would ask the user for information only as it is needed.

Another advantage of this approach is that the system could gradually be interfaced with other parts of the process planning system as they are developed. For example, suppose it is decided to use a GT code describing the parts to be manufactured. Then the process planning system could be instructed that before asking the user each of its questions, it should first check to see if the answer could be deduced from the GT code.

Suppose further that a computer aided design system is later acquired or developed. Then the process planning system could be instructed to direct some of its questions to the computer aided design system. Modules to answer other questions could gradually be added to automate the process selection task as much as possible.

3.2. Representation of Process Planning Knowledge

At present, it is not obvious what all of the factors are that might influence the process selection and sequencing for a particular part, or what effects these factors might have. It may be necessary to make extensive modifications and additions to the decision logic as the system is developed. The system may need decision-making knowledge which can only be gained by talking to a human expert, as was the case with medical diagnosis systems such as MYCIN [10], CASNET [22], or MDX [4].

AI knowledge representation techniques can provide an appropriate way to handle these problems. If implemented correctly, such techniques can allow pieces of knowledge easily to be added, changed, or removed from the system without having to modify other pieces of the system to accommodate each change. Such an extensible system would have distinct advantages. Initially, knowledge could be encoded into the system to allow it to do process selection on some simple class of parts (such as lathe-turned parts). This would give the system developers some experience with the problems involved with automating the process planning task. Once working, the system could have its knowledge augmented to allow it to handle other parts.

In the case of process planning, the problem solving knowledge could be organized as follows. For each type of surface, there would be a list of processes capable of creating the surface. For each process in this list, there would be a frame specifying the restrictions on the capabilities of the process, and the preconditions that must be satisfied before the process can be performed.

In Tables 3, 4 and 5 are possible frames for three processes capable of creating a hole: twist drilling, spade drilling, and rough boring. The

Table 3.
A Frame for a Twist Drill Process to Create a Hole H.

frame for a process to create a hole H
process name: twist-drill

list of restrictions:
1. diameter(H) > .063
   COMMENT: diameter(H) refers to the value in the diameter slot of the frame for the hole H. The meanings of the other functions below are analogous.
2. diameter(H) < 2
3. length(H) < 12 * diameter(H)
4. pos-diam-tol(H) > (.007 * diameter(H)) * * 0.5 + .003
5. neg-diam-tol(H) > (.007 * diameter(H)) * * 0.5
6. loc-tol(H) > .008
7. straightness (H) > (.0005 * length(H)/diameter(H)) * * 3 + 0.002
8. straightness(H) > (.001 * length(H)/diameter(H)) * * 3 + 0.003
9. roundness(H) > .004
10. surface-finish(H) > 100

list of preconditions:
1. either one of the following:
   1a. list of preconditions:
       type (on-surfaceH)) = flat-surface
       x-angle(on-surfaceH)) = x-angle(H)
       z-angle(on-surfaceH)) = z-angle(H)
   1b. spot-face F
       hole(F) = H
2. hole-workspace C
   x-location(C) = x location(H)
   y-location(C) = y location(H)
   z-location(C) = z location(H)
   x-angle(C) = x-angle(H)
   z-angle(C) = z-angle(H)

system would contain similar frames for other hole-creating processes, such as finish boring, rough reaming, and finish reaming.

In the frames shown in Tables 3, 4 and 5, the hole to be created is referred to by the variable name H. Each of the processes requires the existence of other machined surfaces or geometrical entities. These are referred to by other variable names, such as C, S, and H'. The data used in the process restrictions were taken from [5], pp. 164-5.

In the system, the frames for the processes would be used in a manner similar to the production rules for expert computer systems such as MYCIN [10] or KMS [21]. This is illustrated in the sample dialog given in the following section.

3.3. The Control Structure

The following is an example of how a computerized process planning system might work using the knowledge representation techniques described above. We assume that our hypothetical system has a graphics display terminal whose screen is divided into three parts: a display of the workpiece, a display of the surface being currently described by the user, and a space in which the dialog between the user and the system is typed.

At the top level, the control structure for the system is a simple loop:

```
LOOP
        s: = ask-user-for-surface( )
        IF s is a command rather than a surface
        THEN execute s
        ELSE find-way-to-produce(s)
REPEAT
```

If we list the process frames for a surface, the process frames relevant to the preconditions for these frames, and so forth, the resultant structure is a problem reduction graph (see Fig. 2). The subroutine "find-way-to-produce" is a procedure which starts at the top of this graph and calls itself recursively to do a top-down, depth-first search through the graph. Searching this graph will yield a sequence of processes capable of generating the surface, if such a sequence exists.

Suppose a user is designing a workpiece which has a flat horizontal surface with a hole in it. Suppose the flat surface has already been handled by the system, and the user decides to enter the information about the hole. Then the dialog between the system and the user might proceed as outlined in the following paragraphs.

SYSTEM: "Enter next surface."
USER: "hole"

The system retrieves and copies the frame for holes. The new frame, which represents the hole to be produced, is given the name HOLE-1. Using default values for various slots in the frame, the system draws a picture of HOLE-1 in the "current surface" portion of the screen.

SYSTEM: "The hole is called HOLE-1."

"find-way-to-produce(HOLE-1)" is then called. This causes the system to retrieve the frames for all processes capable of creating a hole, and trying to find one whose restrictions are satisfied by HOLE-1. The first process frame retrieved is the "twist-drill' frame. The system checks this frame's restrictions, one by one, to see if they are satisfied.

Table 4.
A Frame for a Spade Drill Process to Create a Hole H.

frame for a process to create a hole H

process name: spade-drill

list of restrictions:

1. diameter(H) > 0.75
2. diameter(H) < 4
3. length(H) < 4 * diameter(H)
4. pos-diam-tol(H) > (.005 * diameter(H)) * * 0.5 + .003
5. neg-diam-tol(H) > (.004 * diameter(H)) * * 0.5 + .025
6. loc-tol(H) > .008
7. straightness(H) > (.0004 * length(H)/diameter(H)) * * 3 + 0.002
8. parallelism(H) > (.0008 * length(H)/diameter(H)) * * 3 + 0.003
9. roundess(H) > .004
10. surface-finish(H) > 100

list of preconditions:

1. one of
   - flat-surface S
     - x-angle(S) = x-angle(H)
     - z-angle(S) = z-angle(H)
   - spot-face F
     - hole(F) = H
2. hole-workspace C
   - x-location(C) = x-location(H)
   - y-location(C) = y-location(H)
   - z-location(C) = z-location(H)
   - x-angle(C) = x-angle(H)
   - z-angle(C) = z-angle(H)

Table 5.
A Frame for a Rough Bore Process to Create a Hole H.

frame for a process to create a hole H

process name: rough-bore

list of restrictions:

1. diameter(H) > 0.375
2. diameter(H) < 10
3. length(H) < 10 * diameter(H)
4. pos-diam-tol(H) > .002
5. neg-diam-tol(H) > .002
6. loc-tol(H) > .0001
7. straightness(H) > .0003
8. parallelism(H) > .0005
9. roundness(H) > .0003
10. surface-finish(H) > 8

list of restrictions:

1. hole H′
   - COMMENT: H′ must be at the same location as H, but the tolerance requirements for H′ are different from those for H.
   - length(H′) = length(H)
   - x-loc(H′) = x-loc(H)
   - y-loc(H′) = y-loc(H)
   - z-loc(H′) = z-loc(H)
   - x-angle(H′) = x-angle(H)
   - z-angle(H′) = z-angle(H)
   - (the frame would also include various restrictions on the diameter, diametric tolerances, locational tolerance, surface finish, straightness, roundness, and parallelism of H′)
2. hole-workspace C
   - (space in which to do boring)
   - x-location(C) = x-location(H)
   - y-location(C) = y-location(H)
   - z-location(C) = z-location(H)
   - x-angle(C) = x-angle(H)
   - z-angle(C) = z-angle(H)

The first restriction is
diameter(HOLE-1) > 0.063.
Whenever the system has no other way of finding out a piece of information, it asks the user.
SYSTEM: "Enter the diameter of HOLE-1."
USER: "1.0"

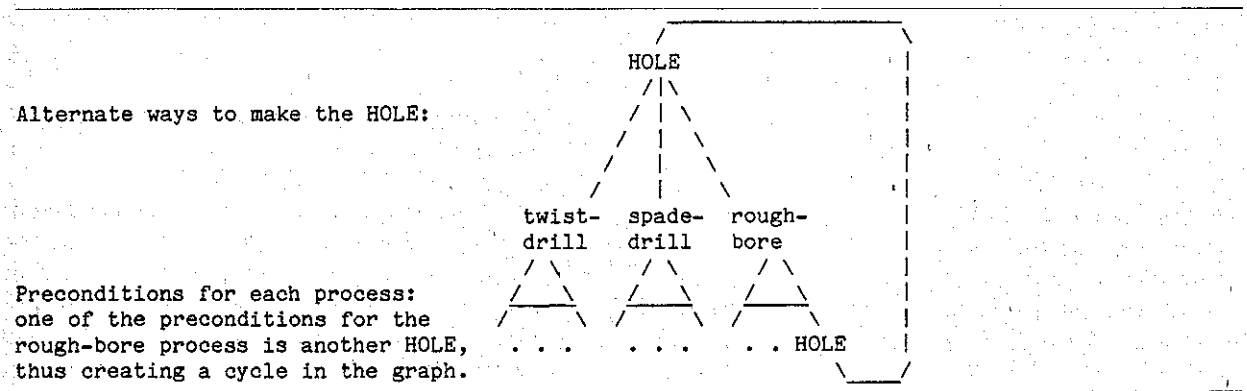


Fig. 2. Problem Reduction for Process Selection.

The value "1" is entered in the "diameter" slot for HOLE-1. Since this changes one of the default values used in the pictorial display of the hole, the picture is redrawn.

Restriction 1 is satisfied, so the system checks restriction 2:
diameter(HOLE-1) < 2.
This restriction is also satisfied, so the system checks restriction 3:
length(HOLE-1) < 12 * diameter(HOLE-1).
For this, the system needs to know the hole's length.
SYSTEM: "Enter the length of HOLE-1."
USER: "1"
Again, the new information is put into the frame for HOLE-1, and the displayed picture of HOLE-1 is redrawn.

Restriction 3 being satisfied, the system checks restriction 4: pos-diam-tol(HOLE-1) > (.007 * diameter(HOLE-1)) * * 0.5 + .003.
SYSTEM: "Enter the positive tolerance of HOLE-1."
USER: "pos-diam-tol = neg-diam-tol = .005" (The user can enter several items at once if he desires, or may even enter arbitrary commands to the system.)

Restriction 4 is not satisfied, so HOLE-1 cannot be created by twist drilling. Thus the system retrieves the frame for another hole-making process: "spade-drill". Using the information already in the frame for HOLE-1, the system ascertains that although the first three restrictions for spade drilling can be satisfied, the fourth one cannot. Thus spade drilling will not work, either.

Continuing its effort to find a usable process, the system retrieves other hole-making process frames, one by one. For each one, it asks whatever questions are necessary to establish whether their restrictions can be met.

The first frame for which all restrictions can be met is the "rough-bore" frame. This means that rough boring can be used to produce HOLE-1, provided that the frame's preconditions are met. As shown in the "rough-bore" frame earlier, there are two preconditions:

1. A hole (H′ in the preconditions) must already exist. However, the tolerances for H′ are not as close as those for HOLE-1).
2. There must be sufficient open space (hole-workspace C in the preconditions) around and above the hole for boring to be done.

Each of these preconditions is set up as a subgoal.

The system works on the first precondition first. Since no hole has yet been created in the workpiece, H′ does not match any existing hole. Therefore, the system creates a new hole frame, HOLE-2, and assigns H′ = HOLE-2.

"find-way-to-produce(HOLE-2)" is now called. The system starts retrieving hole-making process frames and asking questions as before, to determine whether there is a process capable of producing HOLE-2. The slot values for HOLE-2 are not determined by asking questions of the user, but instead are filled in as needed using the information specified in the preconditions for the "rough-bore" process. Since this information refers to slot values in HOLE-1, and since some of these values may not yet be known, the user may be asked questions about HOLE-1.

This time, the system determines that HOLE-2 satisfies the restrictions of the "twist-drill" process, so it starts checking the preconditions for twist drilling.

As shown in the "twist-drill" frame earlier, the first precondition is either (a) that the surface on which HOLE-2 is located be oriented correctly for HOLE-2 to be drilled, or (b) that there be a spot face F for HOLE-2. The surface on which HOLE-2 is located, and its location and orientation relative to this surface are unknown. However, these data can be determined from the corresponding data for HOLE-1. The system now gets this information from the user. By checking the frame it already has for the surface on which HOLE-2 is located, the system ascertains that the surface is oriented correctly for HOLE-2 to be drilled.

Since the orientation and location of HOLE-1 are now known, the system now moves HOLE-1 from its position in the "current surface" portion of the screen to its correct location on the workpiece.

"twist-drill" also requires sufficient space (hole-workspace C in the preconditions) for drilling to be done. The list of applicable processes for creating a hole-workspace contains a process called "ask-about-hole-workspace". The list of restrictions for this process is empty, so the system checks the list of preconditions. In place of a precondition list, the process contains a call to a subroutine which displays an outline of the required space on the screen, and asks the user if

this space intersects the workpiece. The user replies that everything is OK, so this subgoal is satisfied.

The only remaining subgoal is the hole-workspace for the "rough-bore" process. Since this space occupies the same physical volume as the hole-workspace for the "twist-drill" process, the system ascertains that the subgoal is satisfied without having to ask the user any more questions.

At this point, the problem of producing HOLE-1 is solved. The system continues interacting with the user until it has determined how to produce every machined surface on the part to be manufactured, and then it prints out a list of the processes to be used.

4. Summary

We have described how process selection might be done automatically using expert computer system techniques from Artificial Intelligence. This approach is currently being considered for use in the AMRF project at the National Bureau of Standards, and an implementation effort has begun [24]. The main features of the approach are summarized below.

1. The system is interactive. As it needs information about the part to be manufactured it asks the user questions, and it uses the answers to these questions to do process selection. By redirecting some or all of the questions to other computer systems rather than to a human user, the system could be interfaced to other parts of the process planning system as they are developed.
2. The system is organized using AI techniques. Thus the decision-making knowledge can be modified or extended easily, and the system can perform "intelligent" reasoning to do the process selection.

It is hoped that eventually much of the experiential reasoning which goes into other aspects of process planning could be automated the same way, although much of this is beyond the current state of the art. Research along this line is planned by the authors.

Acknowledgements

This work was supported in part by summer research appointments to the authors from the U.S. National Bureau of Standards, in part by research contract N60921-82-C-0083 from the U.S. Naval Surface Weapons Center to the University of Maryland, and in part by research contract NB83SBCA2124 from the National Bureau of Standards to the University of Maryland.

References

[1] J.S. Albus, A.J. Barbera and R.N. Nagel, Theory and Practice of Hierarchical Control, Tech. Report, National Bureau of Standards, Washington, DC, 1980.

[2] Bleich, Computer-Based Consultation, *Amer. Jour. of Medicine 53*, pp. 285–291, 1972.

[3] G. Boothroyd, *Fundamentals of Metal Machining and Machine Tools*, Scripta, Washington, DC, 1975.

[4] B. Chandrasekaran, F. Gomez, S. Mittal and J. Smith, An Approach to Medical Diagnosis Based on Conceptual Structures, *Proc. Sixth Internat. Joint Conf. Artif. Intelligence*, Tokyo, pp. 134–142, Aug. 1979.

[5] T.C. Chang, Interfacing CAD and CAM – A study of Hole Design, M.S. Thesis, Virginia Polytechnic Institute, 1980.

[6] T.C. Chang, The Advances of Computer-Aided Process Planning, Tech. Report, National Bureau of Standards, Washington, DC, Oct. 1981.

[7] T.C. Chang, Group Technology and its Applications: A Tutorial, Tech. Report, Industrial Systems Division, NBS, 1982.

[8] T.C. Chang and R.A. Wysk, An Integrated CAD/Automated Process Planning System, AIIE Transactions 13, 3, Sept. 1981.

[9] R. Davis, H. Austin, I. Carlbom, B. Frawley, P. Pruchnik, R. Sneiderman and J.A. Gilreath, The Dipmeter Advisor: Interpretation of Geological Signals, *Proc. Seventh Internat. Joint Conf. Artif. Intel.*, Aug. 1981.

[10] R. Davis, B. Buchanan, and E. Shortliffe, Production Rules as a Representation for a Knowledge-Based Consultation Program, *Artificial Intelligence 8*, 1, pp. 15–45, 1977.

[11] M.S. Dunn, Jr., J.D. Bertelsen, C.H. Rothauser, W.S. Strickland, and A.C. Milsop, Implementation of Computerized Production Process Planning, Report R81-945220-14, United Technologies Research Center, East Hartford, CT, June 1981.

[12] M.H. Emden and R.A. Kowalski, The Semantics of Predicate Logic as a Programming Language, *J. ACM 23*, 4, 1976.

[13] L.D. Erman, F. Hayes-Roth, V.R. Lesser, and D.R. Reddy, The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty, *Computing Surveys 12*, 2, pp. 213–253, June 1980.

[14] E. Feigenbaum, G. Buchanan, and J. Lederberg, Generality and Problem Solving: a Case Study Using the DENDRAL Program, pp. 165–190 in *Machine Intelligence 6*, ed. B. Meltzer and D. Michie, Edinburgh University Press, Edinburgh, 1971.

[15] C.L. Forgy, The OPS5 User's Manual, Tech. Report, Computer Sci. Depth., Carnegie-Mellon University, 1980.
[16] R.E. Hewgley, Jr. and H.P. Prewett, Jr., Computer Aided Process Planning at the Oak Ridge Y-12 Plant: A Pilot Project, Report Y/DA-8297, Union Carbide, Oak Ridge Y-12 Plant, Oak Ridge, TN, April 1979.
[17] R.D. Holtz, GT and CAPP Cut Work-in-Progress Time 80%, *Assembly Engineering 21*, 7, pp. 16–19, July 1978.
[18] W.S. Mann, Jr., M.S. Dunn, and S.J. Pflederer, Computerized Production Process Planning, Report R77-942625-14, United Technologies Research Center, Nov. 1977.
[19] J. McDermott and B. Steele, Extending a Knowledge-Based System to Deal with Ad Hoc Constraints, *Proc. Seventh Internat. Joint Conf. Artif. Intel.*, pp. 824–828, Aug. 1981.
[20] D.S. Nau, Expert Computer Systems, and their Application to Automated Manufacturing, Tech. Report NBSIR 81-2466, National Bureau of Standards, 1982.
[21] J. Reggia, T. Pula, T. Price, and B. Perricone, Towards an Intelligent Textbook of Neurology, *Proc. Fourth Annual Symposium on Computer Application in Medical Care*, Washington, D.C., pp. 190–199, 1980.
[22] S.M. Weiss, C.A. Kulikowski, S. Amarel, and A. Safir, A Model-Based Method for Computer-Aided Medical Decision-Making, *Artificial Intelligence 11*, 2, pp. 145–172, 1978.
[23] R.A. Wysk, An Automated Process Planning and Selection Program: APPAS, Ph.D. Thesis, Purdue University, 1977.
[24] B. Perricone, Personal Communication, 1983.