Proceedings of the 1999 IEEE
International Symposium on Assembly and Task Planning
Porto, Portugal - July 1999

PROCESS PLANNING USING A GENETIC ALGORITHM APPROACH

João Rocha, Carlos Ramos*, Zita Vale+*
**Departamento de Engenharia Informática* *+Departamento de Engenharia Electrotécnica*
Instituto Superior de Engenharia do Porto
Rua S. Tomé, 4200 PORTO, PORTUGAL
Email: jrocha@ipp.pt

Abstract

This paper presents a system for Computer Aided Process Planning (CAPP), using a precedence graph representation of the operations and their restrictions. Description of machine and tool selection is described, as well as operation sequencing.

A genetic algorithm is used to generate the sequence of operations and to select the machine and tools that minimize some criteria (in the examples, machining time is used).

1. Introduction

Process planning involves the preparation of a plan to describe operations and their sequence, machines, tools and the parameters needed to transform one or more components into a final product [1]. The automatization of process planning, named as CAPP (Computer Aided Process Planning), represents the link between design (CAD – Computer Aided Design) and manufacturing (CAM – Computer Aided Manufacturing) in a CIM (Computer Integrated Manufacturing) system.

Process planning determines the detailed manufacturing requirements for transforming a raw material into a completed part, within the available machining resources.

More global, complex and competitive manufacturing systems markets imply a need for frequently redesigning products and reprogramming. For example, to support new manufacturing technologies such as DFM (Design For Manufacturing), the best process plan for a given part must be generated and fed back to the designer for evaluation.

However, manufacturing systems reprogramming is a complex task involving a considerable set-up time. Most of this time is spent in defining how the new product will be manufactured and in programming the different machines of the manufacturing system (e.g. CNC, robots, AGV's, conveyors, ...). Increasing of system's flexibility is very important to achieve efficiency and productivity improvements.

In any CAPP system, selection of the machining operations sequence is one of the most critical activities for manufacturing a part and for the technical specification in the part drawing. When a sequence is achieved some criteria to determine its fitness should be provided. For example, the goal could be to generate the sequence that minimizes production time or machine utilization or production cost.

Authors previous work in this area using precedence graphs and some traditional search methods similar to A* is described in [2-3]. This paper presents another approach to the process planning problem, using a Genetic Algorithm (GA) to sequence the operations.

Constraints imposed by precedence relations between operations are considered using a precedence graph formalization as well as selection of operations, machines and tools. The duration of the operations, setups, machine and tool changing are also considered in order to determine the production time.

2. Background

"Planning refers to a deliberate behavior, a carefully weighted scheme for action towards some goal" [4].

Although several CAPP systems have been developed, only a few have considered the optimization of operation sequence and the alternative sequence of operations or process plans and, most of them, focus on generating the optimal plan for individual features (geometric forms, such as slots and holes).

To achieve an optimal plan against a predetermined criterion, such as minimum cost, operation selection and operation sequencing must be carried out simultaneously.

Operation selection refers to the determination of all the operations required for each feature. Operation sequencing refers to determination of the execution sequence of all operations, maintaining the precedence relations among all the operations [5].

Recently, research has focused on process planning optimizations, by considering some of the operation selection or sequencing concurrently [6-8].

The potential for using graph theory for operation sequencing is addressed on [9] and [10].

When precedence graphs are used, it is possible to eliminate infeasible machining operation sequences. Some works reported the use of Hamiltoninan path analogies for the process planning problem, based on precedence graph and operation cost matrix [3, 8].

Since planning is a NP-hard problem [11], some global search techniques must be applied. Recently, Genetic Algorithms (GA) have been applied to these problems. The role of GA in combinatorial optimization problems is discussed in [12].

GA are a means of solving problems without explicitly stating the solution. This is done using the idea of *survival of the fittest*. Good solutions are combined with each other to, hopefully, create a better solution. The major genetic operators that reflect nature's evolutionary process are reproduction, crossover and mutation.

The application of GA to process planning requires a method for representing a plan as a string whose elements define a list of machining operations. Each operation can be associated to a machine and tool. The design of appropriate crossover and mutation operators also plays a major role for the successful implementation.

Some works report the application of GA to process planning [13-15] but usually they consider a limited machining environment or a sequential decision making.

In our previous work [3] we intended to address these problems using a classical approach. The work presented in this paper pretends to solve the same kind of problems, but using a GA approach.

3. Computer Aided Process Planning

3.1. Introduction

CAPP could be described as the process of converting a design to a set of manufacturing processes, using computers.

In a CAPP system some factors are usually considered, such as product costs, processing time, setup time, machine groups, selection of operations, machine adjustment data and plan editing. While these do not represent all possibilities considered within a CAPP system, they do indicate the features that are of importance to industry.

3.2. Operations

In this paper operations are defined as instantiations of operators that are the elements that can be used to change the state of the "world" (in this case, the part being manufactured). Once an operation is applied, the state of the "world" changes.

Some operators are able to transform parts or change their shapes (e.g. drilling or milling) and others are able to join parts (e.g. welding or assembly). This means that after an operation we may have a transformation of the parts handled by the operation.

From a process planning perspective, an operation can be identified as an enhancement of the shape, surface finish and size, produced by a countable set of specific physical actions.

3.3. Precedence Relations and Graph Representation

The general goal we want to achieve consists of a set of subgoals. We will use operators to achieve these subgoals. The sequence of operations for the task, together with their association with a machine and a tool is the symbolic plan for the task.

However, the subgoals cannot always be achieved in any order. Often the accomplishment of some goals is restricted by constraints that represent interference among subgoals (known as negative interactions).

For manufacturing tasks, the main constraints are the process constraints, feasibility constraints and geometric constraints. The constraints imply precedence relationships to guarantee that operations will be executed in the correct order and the final operations sequence (plan) is feasible.

It is important to acquire knowledge about these constraints. Sometimes, knowledge about feasibility and geometrical constraints can be automatically acquired with some geometric reasoning capability [16].

Precedence graphs are used to represent the precedence relations between operations. In precedence graphs, each node represents an operation and each path a constraint among operations. Since all the constraints are considered in the graph design, only feasible sequences of operations could be generated from the graph.

3.3. Precedence Cost Matrix

Since detailed information on tools and machining parameters is not available before the plan is determined, the total machining time cannot be used for plan evaluation. Instead a precedence cost matrix is used considering the operations precedence (to consider only valid transitions) and all the costs involved - machine change, tool change and setup cost as well as machining and tool cost.

Machining and tool cost are automatically calculated, since information on geometry of parts can be obtained from a CAD system.

The other costs can be determined from the system configuration and from the machine characteristics.

4. Genetic Algorithm

4.1. Introduction

GA are a family of computational models inspired by evolution. These algorithms encode a potential solution to a specific problem on a simple chromosome-like data structure and apply recombination operators to these structures so as to preserve critical information. GA are often viewed as function optimizers, although the range of problems to which GA have been applied is quite broad [17].

In GA, a candidate solution (a point in the search space) is represented by a sequence of data known as chromosome or string. Codification of data is domain dependent and plays an important role on algorithm performance. A set of chromosomes is called a population and the population at a given time is a generation.

An implementation of a GA begins with a population of (typically random) chromosomes. The chromosome ability to solve the problem is determined by its fitness function, allowing decision on whether one solution is better than another is. Reproductive opportunities are allocated in such away that those chromosomes that represent a better solution to the problem are given more chances to "reproduce" than those chromosomes that are worst solutions.

When all decisions about reproduction were made, crossover operators are applied, taking two parents and combining them to create two new solutions. The two children solutions will replace two previous solutions, thus keeping the population size constant.

To maintain some degree of genetic diversity, minimizing convergence to local solutions of the search space, the mutation operator is applied. This operator makes a small change to the children randomly.

The chromosomes resulting from the operations form the next generation's population. The process is repeated until a predetermined number of generations is reached or until a good solution, according some criteria, is found.

The number of strings in the population will have an effect on how fast the GA can find a solution. With a small population, a good solution will be found quickly, with a large population, it will take longer to find a good solution, but it will have a better chance of finding a great, or the best, solution.

4.2. Chromosome representation

Binary coding has been the usual individual representation in GA for a long time. Binary strings are sufficiently general but they are not always the more natural or the more adequate representation.

When solutions for hard combinatorial problems, like a process planning problem, using GA are considered, representation issues arise. In [18] a discussion of genetic representation issues can be found.

Since knowledge dependent representations could give good results, a string representation similar to the one proposed in [19] is used.

Op_1	Op_1	...	Op_1
M_1	M_2	...	M_3
T_1	T_2	...	T_3

Figure 1 – String representation

The number of *positions* in each string is equal to the number of operations. Each *position* contains a reference to the operation and to the machine and tool used for that operation. Since all combinations of these items can be represented, all possible sequences can be generated thus covering all solution space. An example can be found in Figure 1, where n is the number of operations, Op_i is operation i, M_i is the machine used for Op_i and T_i is the tool used to perform Op_i on machine M_i.

4.3 Initial population

The generation of the initial population in GA is usually done randomly. But, since we use precedence relations between operations, the initial population must consist of strings of valid sequences, satisfying all precedence relations.

In our previous work [3] we describe an algorithm to quickly generate one feasible plan (sequence of operations). The algorithm to generate the initial population will be:

repeat
  randomly select one operation among those who have no predecessors
  generate one feasible plan starting with that operation, choosing next operation randomly, from between those valid
until the initial population size is reached

4.4 Fitness evaluation

The objective of the process planning problem is to minimize the production cost (in terms of time) for the given problem that can be calculated from:

$$PC = \sum_{i=1}^{n} (OC_i + SC_{i+1} * \Omega(OMT_i, OMT_{i+1})) \quad (1)$$

where PC is the production cost, n is the number of operations, OMT is a combination of operation – machine -

tool (an operation is machined on a machine with a given tool), OC_i is the machining cost for OMT I and SC_{i+1} is the setup cost for changing for OMT i to OMT $i+1$.

$$\Omega(x, y) = \begin{cases} 1, & if\ x \neq y \\ 0, & if\ x = y \end{cases} \quad (2)$$

The operation machining cost for a given combination of operation, machine and tool can be automatically calculated [20]. The setup cost is calculated as a sum of following costs: machine setup, tool setup, machine change and tool change.

Only valid OMTs are represented on the matrix. One matrix position has a value only if the precedence relations allow the transition between the two corresponding OMTs.

The fitness function is calculated for each string in the population as described in (1).

4.5 Reproduction

The "roulette wheel" method with "elitism" is used for the reproduction of the strings. This is done by first copying the best member of the population (the one with lowest fitness value) to the following generation and then applying the "roulette wheel" method to the remaining strings.

Although elitism may increase the speed of domination of a population by a super individual, in general it improves the genetic algorithm performance [21].

4.6 Crossover

The crossover operator is applied, at a given probability, to the strings that resulted from reproduction.

Since precedence graphs are used, the crossover operator must ensure that precedence relations are maintained and that only feasible strings are generated.

A variant of the operator proposed by [22] is used. The algorithm is the following:

- based on the string length, two crossover points are randomly generated to select a segment in one parent between these crossover sites;
- the offspring, child 1, is generated by arranging the elements of the selected segment in this parent according to the order in which they appear in the other parent with the order of the remaining elements being the same as in the first parent.
- the role of these parents will then exchange in order to generate another offspring, child 2.

Here some problem-specific knowledge is incorporated. Since the representation of each gene is done using an OMT, it is possible to try an optimization at this point, since it is possible to minimize the machine and tool changes.

Then after applying the crossover operator the following algorithm is used:

```
i = 1
repeat
    if the operation i + 1 has, as an alternative, the
      same machine than operation i
        assign current machine to the operation i + 1
        if the operation i + 1 has, as an alternative,
          the same tool than the operation i
            assign current tool to the operation i + 1
    i++
until i = number of operations – 1
```

An illustration of the crossover operator can be found in Figure 5.

4.6 Mutation

The mutation operator is applied with a small probability to introduce some genetic diversity, thus avoiding being struck at a local optimum. Once again it is important that the application of the mutation operator generates only feasible strings. Three mutation operations were used.

The first one refers to the string and consists on the random exchange of two genes of the string with a predetermined probability. A feasibility test is applied in order to guarantee that the string is feasible and the mutation is applied only if this test returns true.

The other two mutation operators refer to machines and tools and they are similar. The algorithm is the following: randomly select a position in the string with a predetermined probability and randomly choose a machine (or tool) from all the alternatives to replace the current machine (or tool).

An illustration of these operators can be found in Figure 6.

5. An example

Let us consider an example for making a pawn, as shown in Figure 2, starting with an aluminum cylinder. The sequence of operations will be obtained by the feasibility and geometric constraints between parts and tools. There are also some processing constraints.

The operations and their costs (in terms of time) are defined in Table 1. The OMT representation was used → [operation(area), machine, tool], where *area* refers to the areas defined on Figure 2. These costs can be automatically obtained, as described in [22].

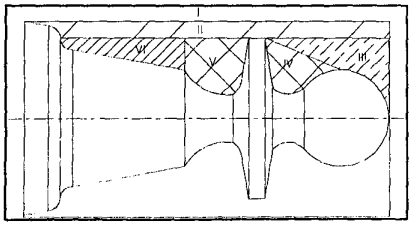

Figure 2 - An example

Table 1 - Duration of the operations (with machines and tools allocated)

N.	Operation	Duration (sec)
1	[remove(I), 2, 1]	60
2	[turning(II), 2, 3]	74
3	[facing(III), 2, 1]	161
4	[facing(IV), 2, 1]	122
5	[facing(V), 2, 1]	172
6	[turning(VI), 1, 3]	77

However, we do not know in which order the operations will be carried out. To know this, we will generate all feasibility, geometric and processing constraints. They are represented on the precedence graph shown in Figure 3, together with the list of valid machine and tools for each operation, represented on Table 2.

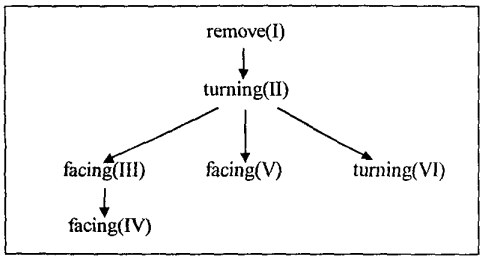

Figure 3 - Precedence graph

An example of the precedence cost matrix (PCM) is represented on Table 3, where *na* means not applicable - i.e. it is not possible to make a transition from one operation to the other one, because some constraints apply. A complete PCM will include all valid OMTs and the corresponding (if applicable) setup costs.

String No.	String	Fitness value
1	1, 2, 3, 5, 6, 4	707
2	1, 2, 6, 5, 3, 4	738

Figure 4 – Example of representation and fitness

Examples of valid strings are represented on Figure 4, where all the operations needed for machining the pawn are included and all the precedence relations are considered.

Table 2 – Valid machines and tools

N.	Operation	Machines	Tools
1	remove(I)	1, 2	1, 2, 3
2	turning(II)	1, 2	2, 3
3	facing(III)	2	1
4	facing(IV)	2	1
5	facing(V)	2	1
6	turning(VI)	1, 2	2, 3

Table 3 - Setup times (from one operation to another)

	1	2	3	4	5	6
1 - [remove(I), 2, 1]	-	3	na	na	na	na
2 - [turning(II), 2, 3]	na	-	3	na	3	34
3 - [facing(III), 2, 1]	na	na	-	1	2	35
4 - [facing(IV), 2, 1]	na	na	na	-	1	34
5 - [facing(V), 2, 1]	na	na	1	1	-	33
6 - [turning(VI), 1, 3]	na	na	34	33	33	-

In Figure 5, an example of crossover operator application is presented. The child 1 and the problem-specific algorithm application are exemplified. The numbers refer to the operations numbers on Table 3.

a) string crossover

Parent 1	Parent 2	Cross points	Child 1
1, 2, 3, 5 6, 4	1, 2, 6, 5, 3, 4	2, 5	1, 2, 5, 3 6, 4

b) machine change

before (2 machine changes – 2→1→2)

Operation	1	2	5	3	6	4
Current machine	2	2	2	(2) → 1 →	2	
Alternative machines	1	1	-	-	(2)	-

after (0 machine changes)

Operation	1	2	5	3	6	4
Current machine	2	2	2	2	2	2
Alternative machines	1	1	-	-	1	-

Figure 5 – An example of crossover application

In Figure 6 an example of string mutation and machine application is presented. After the production of the offspring and the application of the operators, a new generation is obtained. The whole process is repeated for a specified number of generations. When this number is reached, the string corresponding to the lowest fitness value is taken as the optimal operation sequence.

a) string mutation

String	Resulting string	Feasible?
1, 2, 5, 3, 6, 4	1, 2, 5, 6, 3, 4	Yes

b) machine mutation

before

Operation	1	2	5	3	6	4
Current machine	2	2	2	2	1	2
Alternative machines	1	1	-	-	2	-

after

Operation	1	2	5	3	6	4
Current machine	2	1	2	2	1	2
Alternative machines	1	2	-	-	2	-

Figure 6 – An example of mutation application

6. Conclusions

This paper presents a GA approach to process planning, with the following characteristics:

- users can define and modify the manufacturing environment
- since all available machines and tools are considered, the whole solution space is considered, allowing globally optimal solutions to be found
- only feasible sequences are considered and generated
- new crossover and mutation operators are used

7. References

[1] A. Kusiak; *Intelligent Manufacturing Systems*, Prentice-Hall International Series in Industrial and Systems Engineering, 1990

[2] J. Rocha, C. Ramos and Z. Vale; "Sequencing Operations for Process Planning"; in *Proc. Practical Applications of Prolog - PAP'95*, 1995

[3] J. Rocha, C. Ramos and Z. Vale; "Representing and Generating Operation Sequences for Manufacturing Tasks"; *International Symposium on Assembly and Task Planning - ISATP'97*; pp. 134-139, August 1997

[4] M. Ghallab and A. Milani; *New Directions in AI Planning*; IOS Press, pp. V., Frontiers in Artificial Intelligence and Applications, vol. 31, 1996

[5] F. Zhang, Y. F. Zhang, A. Nee; "Using Genetic Algorithms in Process Planning for Job Shop Machining"; *IEEE Tran. on Evolutionary Computation*, vol. 1, no. 4, pp. 278-289, November 1997

[6] C. Hayes; "Automating process planning: using feature interactions to guide search"; *Journal Manufacturing Systems*, vol.8, no. 1, pp. 1-14, 1990

[7] C. Chu and R. Gadh; "Feature based approach for set-up minimization of process design for product design"; *Computer Aided Design*, vol. 28, no. 5, pp. 321-332, 1996

[8] S. Irani, H. Koo and S Raman; "Feature based operation sequence generation in CAPP"; *International Journal of Production Research*, vol. 33, no. 1, pp. 17-39, 1995

[9] R. Sundaramn; "Process planning and machining sequence"; *Computers and Industrial Engineering*, no. 11, 184-188, 1986

[10] P. Prabhu et al; "An operations network generator for computer aided process planning"; *Journal of Manufacturing Systems*, no. 9, pp. 283-291, 1990

[11] S. Gupta and D. Nau; "Optimal Block's World Solutions are N-P"; *Technical Report*, Computer Science Department, University of Maryland, 1990.

[12] D. Goldberg; *Genetic Algorithms in Search, Optimization and Machine Learning*, Addision-Wesley, 1989

[13] J. Váncza and A. Márkus; "Genetic algorithms in process planning", *Computers Industry*, vol. 17, pp. 181-194, 1991

[14] B. Awadh, N. Sepehri and O. Hawaleshka; "A computer-aided process planning model based on genetic algorithms"; *Comput. Oper. Research*, col. 22, no. 8, pp. 841-856, 1995

[15] D. Yip-Hoi and D. Dutta, "A genetic algorithm application for sequencing operations in process planning for parallel machining"; *IIE Transactions*, col. 28, pp. 55-68, 1996

[16] C. Ramos and E. Oliveira; "CIARC: A Multi-Agent Community for Intelligent Assembly Robotics Systems with Real-Time Capabilities"; *Proc. of IEEE Int. Conf. on Systems, Man and Cybernetics*, 1993

[17] D. Whitley; "A Genetic Algorithm Tutorial"; *Technical report CS-93-103*, Department of Computer Science, C. S. U., November 1993

[18] Z. Michalewicz; *Genetic Algorithms+Data Structures = Evolution Programs*, Springer Verlag 1996

[19] R. Bruns; "Direct chromosome representation and advanced genetic operators for production scheduling"; Proc. Fifth Int. Conf. On Genetic Algorithms, pp. 352-359, 1993

[20] J. Rocha et al; "Process and Execution Planning for Manufacturing Systems"; to appear in *Proc. of Int. Symp. Assembly and Task Planning* - ISATP'99; 1999

[21] L. Davis; *Handbook of Genetic Algorithms*, Int. Thomson Computer Press, 1996

[22] S. Reddy, M. Shunmugam and T. Narendran; "Operation sequencing in CAPP using genetic algorithms"; *Int. J. Prod. Res.*, vol. 37, no. 5, pp. 1063-1074, 1999