Proceedings of the
2000 Japan•USA Flexible Automation Conference
July 23-26, 2000, Ann Arbor, Michigan

2000JUSFA-13194

OPERATION CLUSTERING IN PROCESS PLANNING FOR RECONFIGURABLE MACHINING SYSTEM DESIGN

Catherine Ling, Derek Yip-Hoi, Yoram Koren
Department of Mechanical Engineering and Applied Mechanics
University of Michigan
Ann Arbor, MI 48109
lingm@engin.umich.edu; yiphoi@engin.umich.edu; ykoren@engin.umich.edu

Keywords: Reconfigurable Machining System, Operation Clustering, System Configurator, Process Planning, Dedicated Machining Lines, Flexible Manufacturing Systems

ABSTRACT

A systematic methodology for automatic machining operation clustering is an important component in *Reconfigurable Machining System* (RmS)[1] design. The task of designing a RmS is performed by a *System Configurator*, of which *the Decision Support System for Machine Selection* is one major component. Machining operation clustering is the first step in machine selection. One important form of clustering identifies sets of operations that have the potential to be machined in parallel. Such *parallelism-based operation clusters* as we refer to them, must satisfy a set of constraints in order to be processed simultaneously on a gang spindle head. Minimum feature spacing is one major mechanical constraint due to bearing size limitation that must be considered. This paper first proposes a model for developing a *Decision Support System for Machine Selection*, then presents a patterning algorithm for obtaining identical parallelism-based clusters that satisfy the minimum feature spacing constraint where applicable. The algorithm obtains identical clusters by first searching for translational vectors and then extracting the appropriate end-points. The algorithm automatically obtains all alternative solutions so that finding identical patterns on different faces of the part or on different parts within the target family can be implemented. This strategy maximizes the usage of identical spindles and/or machines. The cost associated with redesigning, testing, and reconfiguration is significantly reduced.

INTRODUCTION

Many manufacturing industries currently utilize a portfolio of *dedicated manufacturing lines* (DML) and *flexible manufacturing systems* (FMS) to produce their products. Dedicated machining lines achieve high productivity but may result in a large amount of idle capacity if large fluctuations in product demand occur. Flexible manufacturing systems, on the other hand, produce a variety of products on the same system As discussed in [Koren 1999], a FMS is expensive because it consists of general purpose machines that can produce any part from a large part family. Addressing the new market conditions of large fluctuations in product demand and short product life cycle requires a new type of manufacturing system that achieves both high flexibility and high batch-type productivity. *Reconfigurable Machining Systems (RmSs)* are one option for such a system. The main components of RmSs are CNC machines and Reconfigurable Machine Tools (RMTs)-a new type of machine that has a modular structure enabling the reconfiguration of its resources (e.g., adding a second spindle unit) according to the need. A RmS is defined as [Koren 1999] as:

> *A machining system designed at the outset around a part family and for rapid adjustment of its production capacity and functionality by changes of its whole structure as well as its hardware and software components.*

We refer to the complete methodology for designing and analyzing a RmS as the *System Configurator*. Figure 1 shows a flowchart representing the activities and interconnections within the System Configurator. There are two major blocks of tasks as shown: the *System Level Process Planner* and *Configuration Evaluation Tools*. In this paper we are concerned with the first block of tasks and in particular the 3rd activity, *A Decision Support System for Machine Selection*. For more details on the System Configurator model, we refer the reader to [Ling et al. 1999].

Operation clustering is a key step in determining the machines to be used to produce a target part family. This activity decides how the set of operations required to produce

[1] In our research we use small 'm' to distinguish the term 'Reconfigurable Machining System' (RmS) from the more general term 'Reconfigurable Manufacturing System' (RMS).

the final part, are to be divided up amongst the machines that will comprise the system. Clustering must take into account part tolerances and parallelism. One application of parallelism is the use of gang spindle heads found on many dedicated systems. These heads remove the need for time consuming stitching moves using a single spindle, as well as tool changes. Identifying appropriate spindle patterns for these heads is an important design issue. Mechanical constraints (such as minimum bearing sizes) can restrict the formation of a single pattern. In such cases, identifying repeated patterns can reduce design costs significantly. When common repeated patterns can be identified on different parts within the part family around which a RmS is designed, this can lead to reductions in the reconfiguration costs.

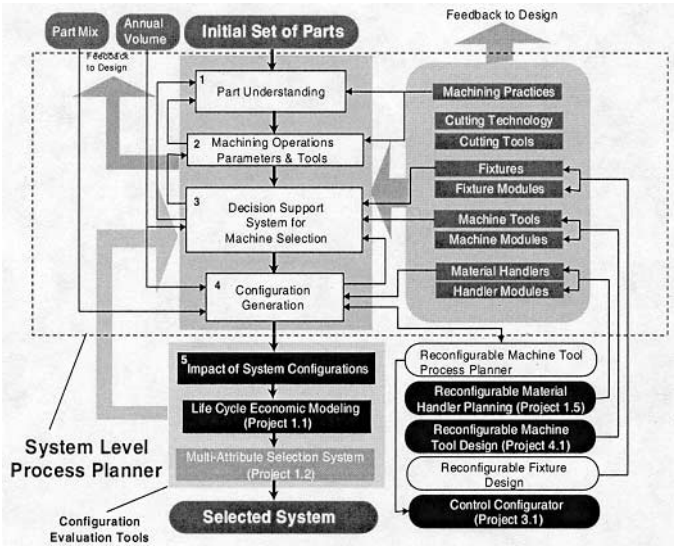


Figure 1. System Configurator for RmS

This paper presents initial work in developing design tools that can assist the system designer in identifying repeating patterns. In the following section we will discuss the *Decision Support System for Machine Selection* methodology to provide a context for the research on pattern identification. We will follow this by a discussion of the cases that the patterning algorithms need to be able to handle, and of an approach for one of these cases with some results from a prototype implementation. The final section will discuss our planned future work in this area.

DECISION SUPPORT SYSTEM FOR MACHINE SELECTION

In the FMS domain, machine tool builders provide general purpose CNC machine tools, with which manufacturers perform process planning. RmS design poses an inverse problem. The concept of customized flexibility, or designing for a part family is the essence of RmS design. In the RmS domain, the part family is given first and machines should be designed around the manufacturing requirements of the part family instead of being general purpose. Methodologies that achieve concurrent process planning and machine selection should be developed. Moreover, RmS design benefits from the identification of common operations across the entire part family through the reduction of reconfiguration cost. Assuming that machining features and operations have been identified, the solution process comprises three major steps as shown in Figure 2: *Single Part Operation Clustering*, *Multi-Part Operation Clustering*, and *Setup Planning and Machine selection*. We will discuss each of these briefly in the following sections.

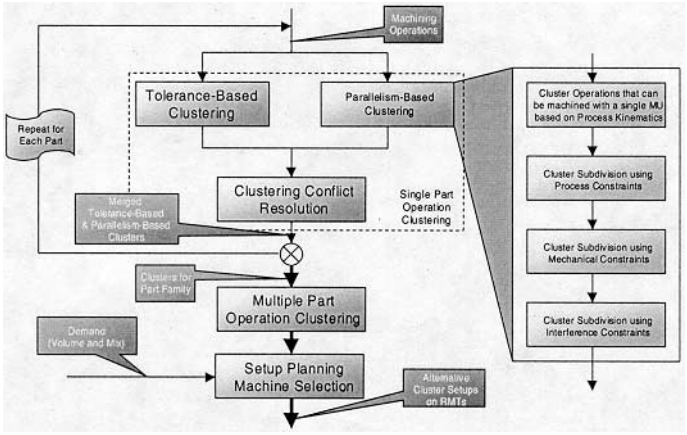


Figure 2. A flow chart for concurrent process planning and machine selection.

Single Part Operation Clustering

In the CNC domain, the grouping criteria for single part operation clustering strongly depends upon the assumed machine configuration. Most of the previous research performed in this area focuses on: (i) preserving feature tolerance relationships [Boerma & Kals 1988], [Demey et al. 1996], [Delbressine et al. 1993], [Zhang and Lin 1999]; (ii) minimizing the number of setups [Chu & Gadh 1996], [Ozturk et al. 1996], [Sarma & Wright 1996], [Yut and Zhang 1995],[Zhang and Lin 1999]; (iii) minimizing machining time and cost [Sormaz & Khoshnevis 1996]. The above research assumes that the machine platforms are general-purpose CNC machines. Little effort has been made on the problem of the automatic grouping of operations for parallel machining on custom designed machine tools. At the same time, end users of systems and Original Equipment Manufacturers (OEMs) are manually forming operation clusters based on identical feature functionality during the system design process. However, little has been done to verify automatically whether these lead to good operation clusters. Moreover, little research has been done to search for common operation clusters across an entire part family.

In the RmS domain, machines are custom designed and selected (or designed) as part of the solution. Thus, the capabilities of machines can no longer be assumed a priori and

used to derive clustering principles. These principles should be based upon the manufacturing requirements of the part family. Part quality, productivity, and commonality across the part family are in general the main criteria that need to be considered. By clustering together operations that have tight tolerance (tolerance-based clustering in Figure 2), the designer's specifications are satisfied. By utilizing parallel machining concepts (parallelism-based clustering in Figure 2), both the number of active machine tool axes and the machining time can be reduced. By designing a RmS for common clusters of operations across the part family whenever possible (Multiple Part Operation Clustering in Figure 2), reconfiguration cost can be reduced.

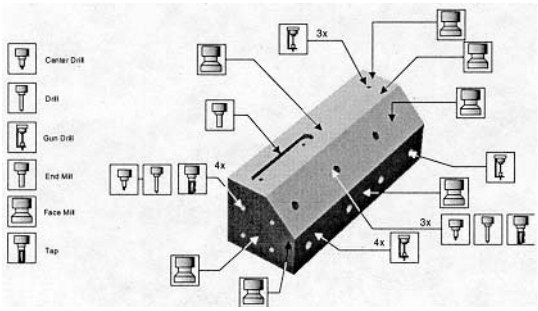


Figure 3. Required machining operations on P

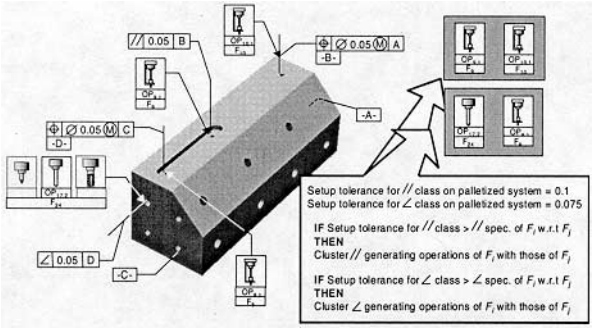


Figure 4. Tolerance-based clustering

For the part shown in Figure 3, Figure 4 shows a tolerance-based clustering example. The rule compares the setup tolerance of a palletized material handling system with tolerance specifications for different features requiring hole machining processes. Based on this rule and the setup tolerance it can be seen that certain operations need to be performed in the same setup if part functionality is to be satisfied. Tolerance-based clustering thus ensures that operations on features such as the ones shown in Figure 3 are always performed on the same machine tool.

Parallelism-based clustering identifies groups of operations that can be machined using cutting tools mounted on a single machining unit and driven by a common set of axes. Typically, the necessary condition for a set of operations to be done in parallel is that they all affect the same face and have an identical approach direction. In addition, the following conditions must be examined:

1. *Similar process kinematics requirements*: Operations of different process types can be grouped only if they require similar process kinematics. For example, drilling and face-milling require a different number of machine axes and so cannot be machined in parallel using the same set of axes.
2. *Process constraints:* Dynamic interaction may occur between operations machined in parallel. These interactions can cause deflections in the cutting tool, machine, and fixture that in turn affects the surface finish. Process constraints need to be examined to ensure that the surface finish of one operation is still within specification under the influence of other operations in the same parallelism-based cluster.
3. *Mechanical constraints*: Machine design mechanical constraints need to be considered in evaluating a proposed parallelism-based cluster. For example, due to bearing size and housing limitations, there is a lower limit on the distance between spindles in a gang spindle head. Therefore, there should be sufficient spacing between features that are processed in parallel. In addition, the size limitation on a gang spindle head might impose a maximum distance constraint among features that are processed in parallel. A third constraint is the power limitation that will impose a constraint on the maximum number of operations that can be processed in parallel.
4. *Interference Constraints*: Interference of tool/holder assemblies and spindles with the workpiece and fixtures may prevent clustering.

An example of a mechanical constraint is shown in Figure 5. Here, preliminary clusters of center drilling, drilling and tapping for processing the four holes violate a minimum spindle distance constraint. The correct parallelism-based clustering must split these operations appropriately to eliminate the violation.

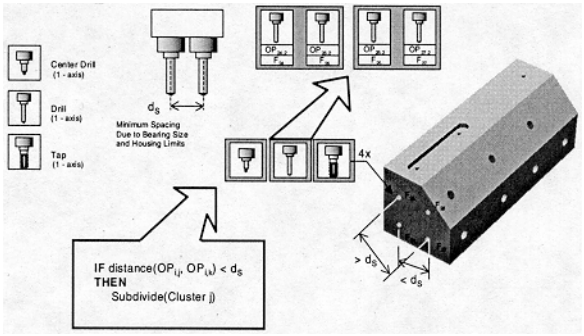


Figure 5. Decomposition using mechanical constraint

Multi-Part Operation Clustering

Multiple Part Operation Clustering groups operation clusters across a part family the parts of which have similar machining

requirements. The goal is to machine each multi-part cluster identified on the same RMT where the reconfiguration capabilities of the RMT are chosen to span the machining requirement variation within that multi-part cluster. The variation in manufacturing requirements of each multi-part cluster determines the amount of reconfiguration required on its corresponding RMT. The higher the similarity among the manufacturing requirements across the part family, the smaller the reconfiguration requirements imposed on the RMT, and the lower the corresponding reconfiguration cost. Thus, the goal of multi-part clustering is to reduce reconfiguration cost through identifying and grouping the most similar sets of single part operation clusters into multi-part clusters.

Group technology has been applied in numerous domains for identifying and grouping entities with the most similarity. While group technology has been applied to part family formation [Han and Ham 1985], [Kusiak 1987], [Selim et al. 1998], and [Sarker 1996], we find no research that addresses the issue of multi-part operation clustering across a part family and its relationship to reconfiguration. While the approaches for part family formation may be extendable to multi-part cluster formation, different coding and similarity measures have to be developed to reflect the attributes of this domain. New metrics need to be established to measure the "distance" between single part operation clusters when forming multi-part clusters. These metrics need to reflect the attributes that contribute to reconfiguration cost. This would include process type, fixturing surfaces, tool access direction, operation patterns (tool diameter, number of operations, spatial distribution of features), location of the operation patterns.

Setup Planning and Machine Selection

This final step determines suitable machine tool configurations for machining the single and multi-part operation clusters determined previously. The two activities involved at this step are (1) the grouping of operation clusters into setups based on fixturing, material handling and productivity requirements, and (2) the selection (or design) of appropriate machine tools to machine these setups.

Cycle time is a major factor to be considered in generating setups. The grouping of operation clusters should take into account the total number of stations required, the complexity and cost of each station, and the system cycle time limit.

A filtering process is adopted for matching a setup's machining requirements with machine tool functionality and capacity. The filtering process will consider machine tool process type, machine tool geometry, spindle power, tooling, machine tool kinematics, and machine tool accuracy. These processes are used to sift through machines in a database and identify the ones with suitable capabilities. Alternatively, these filters can be used to supply constraints to a synthesis tool that will build a machine tool with matching capabilities from modules.

PATTERNING PROBLEM

As discussed previously, parallelism-based clusters have the potential to both reduce the cycle time of a system as well as the complexity of machine tools. However, as also noted these clusters must obey certain constraints. When one or another of these constraints cannot be satisfied, it becomes necessary to separate the operations and form smaller clusters. In doing this, an advantage is to be had by forming these smaller clusters such that they have identical spatial patterns. This reduces the design effort and manufacturing cost for producing a gang spindle head for machining the pattern. Often the same head may be used to complete each pattern by simply translating or indexing the workpiece. If cycle time limits prevent these patterns from being machined in sequence, a second similar head can be used on a duplicate machine that may either be in sequence or in parallel with the first.

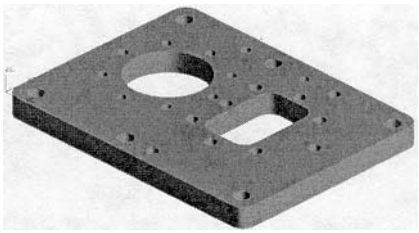


Figure 6. An example part

With a part family, when one of the parts in the family has operations that are best machined by using duplicate pattern clusters, it becomes necessary to consider if other parts in the family have similar clusters. When alternatives exist, choosing the right option becomes critical in minimizing the amount of reconfiguration that needs to take place to the machining system when its converted between parts in the family.

As the part in Figure 6 illustrates, the task of identifying repeating patterns manually is challenging. Even if a pattern can be identified visually, it cannot be assumed that it is correct unless all the relevant dimensions are compared. This motivates the need for developing computer-based tools for pattern identification. The following sections will introduce the mathematical formulation, problem decomposition, and initial work on solution procedures for patterning algorithms for this problem. For simplicity we assume that patterns are for hole type features that can be generated by drilling operations. Patterns for other types of features are possible. However, the kinematics of the operations needed for these features will have to be the same.

Patterning Problem Formulation and Decomposition

The patterning problem in operation clustering can be mathematically formulated as follows:

Given a set S of points (each point corresponding to the center of a hole), find the minimum number of N subsets $s_1, s_2, \ldots, s_N$, where

(1) $\bigcup_{i=1}^{N} s_i = S$

(2) $\forall i, \forall\ p,q \in s_i, d(p,q) \geq d_{\min}$ (d_{min} is a value corresponding to bearing size limitation and *d()* stands for the distance between two points)

(3) $\forall i, j \in [1,N],\ i \neq j$, transform T_{ij} can be found, such that $T_{ij}(s_i) = s_j$

Condition 1 ensures completion of the subdivision, i.e. all points belonging to the original set have been assigned to a subset. Condition 2 guarantees that all points within a subset satisfy the minimum spacing constraints. Condition 3 ensures the formation of repeated patterns.

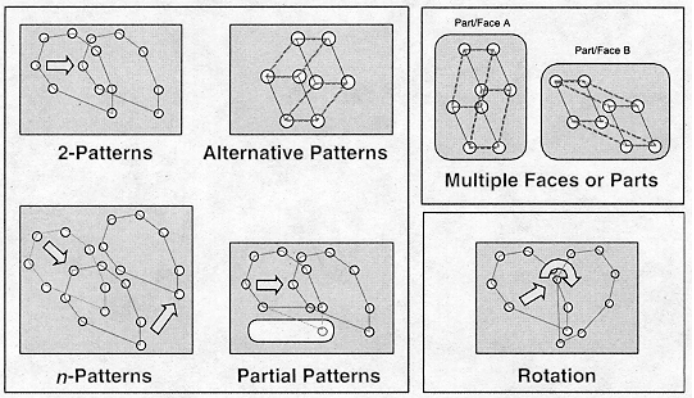


Figure 7. Patterning Problem Category

The general patterning problem can be classified into six categories as shown in Figure 7.

The 2-*pattern* version of the problem applies to the case when only a single face of a part is being considered and the entire group of holes is sub-divisible into two subsets that coincide with each other through pure translation.

The *n-pattern* problem differs from the 2-pattern problem in that the algorithm has to determine the minimum number of subsets *(N)* that the group of holes needs to be divided into as part of the solution.

The alternative pattern problem refers to cases where there are multiple ways of decomposing a group (alternative patterns are shown by different line types).

The partial pattern problem refers to cases where one cluster is a subset of another cluster. Due to the existence of open space due to the geometry of the part or other features, identical gang heads can be utilized even though the two clusters are not identical.

The rotation problem is a generalization of the 2-pattern and n-pattern problems where generalized transformations that include both translation and rotation can exist between subsets.

The multiple face/part problem applies to cases when alternative patterns exist. To further save spindle or machine design costs, the patterns from another face of the part or face from a different part in the part family are considered to generate a solution that is common to multiple parts and faces. In the figure, alternative patterns are shown by different line types on both parts/faces. The pattern that is common to both parts/faces (shown by the solid line) should be selected.

Algorithm

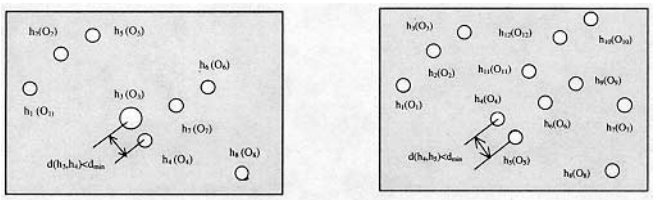
This problem can be tackled in a number of ways. From the graph theory point of view, the patterning problem is combinatorial since it is equivalent to searching a *n*-node graph to find two identical *n/2*-node sub-graphs when considering the 2-pattern problem, for example. An approach through trial-and-error will first randomly select *n/2*nodes to form one sub-graph and group the rest of the *n/2* nodes in another sub-graph, then compare the two sub-graphs to see whether they are identical. For the worst case, $C_{n/2}^{n}$ attempts may be needed before the solution can be found. When the minimum spacing constraint is considered, the two nodes that are violating the minimum spacing constraint should be in different sub-graphs. A trial sub-graph is obtained by selecting *n/2*-1 nodes from *n*-2 nodes. The complexity of the solution process is reduced to $C_{n/2-1}^{n-2}$. This combinatorial problem is NP-complete ([Karp 1972]) by nature and normally requires heuristic search algorithms to obtain approximate results. However, approximate answers are not satisfactory for operation clustering since different gang spindle heads or machine designs are still needed for operation clusters that are even slightly different. Therefore, the graph theory approach is not suitable for solving the patterning problem.

Another point of view, which is more promising, is to look at this problem from a computational geometry point of view and treat each subset as a rigid body. Since all subsets are identical, the same homogenous transform can be applied to transform all member points in any one subset to their corresponding points in other subsets. When a group of points can be transformed to another groups of points under the same transform, then this group of points forms a pattern. This suggests identifying the pattern by looking for a subset of nodes that can be transformed into another subset of nodes under the same homogenous transform. As a first step, we start with the case when only translational transforms exist between subsets. For this case, translational vectors exist between pairs

of their member points. In addition, all translational vectors should be parallel and of the same length. This suggests sub-dividing the set by first identifying all translational vectors or parallel lines, and by then extracting their end-points. We will talk about the details of the approach for the translational case in the next two sub-sections.

2-pattern

For the 2-pattern problem, given that there are n holes (corresponding to n drilling operations) in the two subsets, there will then be $n/2$ translational vectors (i.e., $n/2$ equal length parallel lines). A line is a translational vector if it is one of the $n/2$ parallel lines.



(a) (b)

Figure 8. Original hole pattern and group of operations

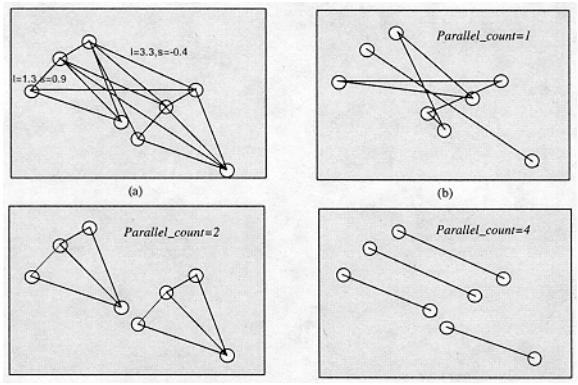
Next we describe the detailed steps of the algorithm for the 2-patterning problem with an example. Figure 8 (a) shows a 2-D pattern of eight holes, two of which (h_3 and h_4) violate the minimum spacing constraint. Therefore, the group of operations on this hole-pattern has to be sub-divided. Effort should be made to recognize identical subsets in the sub-division process. While it is hard to do so by pure observation, a pattern extraction algorithm can successfully recognize the identical subsets.

The algorithm contains the following steps.

(1) Calculate the slope and length of the line segments connecting any two points in the original set: For this example, a total of $M = c_2^8 = 28$ lines are obtained by connecting any two points. Figure 9(a) illustrates some line segments together with their slopes and lengths.

(2) Determine the number of lines that are in parallel with each line segment: This step is accomplished by looping through all line segments and comparing their slopes and lengths. If two lines are of the same slope and length, then their corresponding entries in array *parallel_count* are increased by one. Figures 9(b) through (e) show the result and sample pseudo-code for this step.

(3) Identify translational vectors, i.e. identify the line segments that are one of the $n/2$ parallel lines: This step is accomplished by looping through the entries of all line segments in the array *parallel_count* and identifying the ones with entries being equal to $n/2$ (procedure *finding_line* shown in figure 9(f)). When there are alternative solutions, all translational vectors are identified. They are decomposed according to their slope using procedure *assigning_group* that will be introduced the next section.




```
% Compute parallel_count by looping through line
  segments and compare slope and length
  procedure parallel_count
    For i=1, size(index)
        For j=1,size(index)
            if slope(i)==slope(j) and length(i)==length(j)
            then
            parallel_count(i)<-parallel_count(i)+1
            parallel_count(j)<-parallel_count(j)+1
        End
    End
```

(e)

```
% Identifying line vectors whose parallel_count
  equals to n
  procedure finding_line
    For k=1, nl
        indice<-Find_index (parallel_count(k)==n)
    End
```

(f)

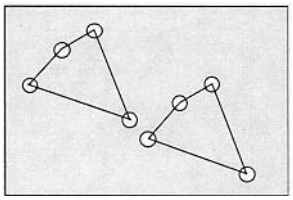
Figure 9. Slope, length, and parallel_count of selected line segments

(4) Form subsets: This step is achieved by collecting end-points of all translational vectors identified previously. Care should be taken in ensuring that end-points on the same side of the vector are grouped together (procedure *identifying_sets*). Figure 10 (a) shows the sample pseudo-code for this task.

Figure 10(b) shows the result where points (center of holes) belonging to the same subset are connected by lines.

```
% Collecting end points of the translational vectors identified
  previously
  procedure identifying_sets
    [first(1),second(1)]<-Find_end_points(index=indice(1))
    signx(1)<-px(first(1))-px(second(1)) {px is the x-coordinate
    of the corresponding line segment}
    For i=1, size(indice)
        [first(i), second(i)]<-Find_end_points(index=indice(i))
        signx(i)<-px(first(i))-px(second(i))
        if different_sign(signx(1),signx(i))==true
        then
        switch(first(i), second(i))
    End
```

(a)



(b)

Figure 10. Subsets identified

n-pattern

For the n-pattern problem, given that there are n holes in the original set and that it can be divided exactly into m subsets, there will then be C_2^m sets of parallel lines with n/m lines in each group. The total number of translational vectors

will be $C_2^m * n/m$ with only the vectors belonging to the same group being parallel to each other. This section describes the adaptation of the 2-pattern method to the n-pattern problem. Two additional tasks are required: (1) to find how many identical subsets the original set should be divided into; (2) to group together parallel translational vectors. Six major steps are required for this problem category. These steps are illustrated through the example shown in figure 8(b).

The first two steps, i.e., calculating the slope and length of the line segments and creating the array *parallel_count*, are the same as the 2-pattern problem. Figure 11 (a) show the entries in *parallel_count* for selected line segments.

The next step is to determine the number of identical subsets. This number equals the total number of points divided by the maximum entry in the *parallel_count* array (the maximum entry equals to the number of parallel_lines in each group, i.e., the number of elements in each subset). For this example, the maximum entry in this array is 4 and the number of subsets is equal to $12/4 = 3$, i.e., three subsets exist.

We then identify translational vectors by searching the entries corresponding to all line segments in the array *parallel_count* and by extracting the ones being equal to *n/m*. Figure 11 (a) shows all translational vectors.

```
% assigning translational vectors into groups
  procedure assigning_group
  For i=1, size(indice)
      If slope(index(indice(i)))=new_slope
      Then group_slope(j)=slope(index(indice(i)))
      j=j+1
  End
  For i=1,size(indice)
      For j=1,number_of_subsets
          If slope(index(indice(i)))==group_slope(j)
          Then j=assign_group(indice(i))
      End
  End
```

(a) (b)

Figure 11. Groups of translational vectors

Next we group parallel translational vectors. For the n-pattern problems, there are C_2^m sets of translational vectors and these must be correctly grouped according to parallelism to ensure the proper formation of subsets. This step is also necessary for the 2-pattern problem when alternative solutions exist. This is achieved by looping through all translational vectors and assigning them into groups according to their slope (procedure assigning_group). Figure 12 (a) shows the result where translational vectors are assigned to three different groups (shown by different line types). The last step "form subsets" is the same as in the 2-pattern approach.

The minimum spacing constraint will be tested again after solutions are obtained. If all solutions satisfy the minimum feature spacing constraint, the one that is common to multiple surfaces or multiple parts should be selected. If only one solution satisfies this constraint, this solution should be selected. If no solution meets this constraint, the patterning algorithm should be applied to each subset to sub-divide further.

For this computational geometry approach, the total number of line segments is C_2^n for a *n*-point set. Since the translational vectors are obtained by comparing a pairs of lines, the number of operations required to search for translational vectors from these line segments is $C_2^{C_2^n}$, which is $O(n^4)$ polynomial complexity. For the n-pattern problem, there is an additional step to group the translational vectors according to their slope, the computational complexity of this operation is of a lower order. Thus, whereas the graph approach is a combinatorial NP-complete problem, a computational geometry approach provides a much simpler problem of polynomial computational effort.

Application Example

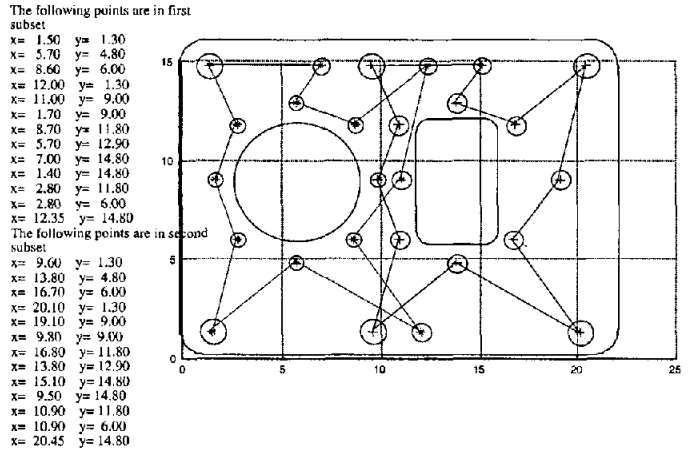


Figure 12. Patterning result

In this section, the patterning algorithm is applied to the part shown in Figure 6. This part has both casting features (the big hole and the pocket) and machined features. Due to a minimum spacing constraint, it is not possible to process all the drilling operations required to produce the machined features using a single gang spindle. Efforts should be made to group these features into identical clusters to save spindle design and part storage cost. While it is hard to do so manually, the desired results can be obtained using the patterning algorithm we have developed.

Figure 12 shows the grouping results using both graphics and text output. Two identical parallelism-based clusters (the center of those holes are marked by '*' and '+" respectively) are generated by the patterning algorithm.

FUTURE WORK

In this paper we have presented a model for concurrent process planning and machine selection and a patterning algorithm for solving operation clustering for translation type patterns. Further research currently underway includes:

- Expansion of the algorithm to general homogeneous transforms including rotation. One approach we are investigating is to adopt the idea of rotational vectors. Similar to the translational problem where translational vectors exist between subsets, for this case rotational vectors (with respect to a fixed center of rotation) exist between subsets. These rotational vectors should correspond to the identical orientation change when the points in one subset are transformed to those in another subset. The subsets can be formed by first identifying the rotational angle for transformations that map lines of the same length onto each other. By identifying groups of mappings with similar angles in a similar manner to the parallel vector groups for the translational problem, general patterns can be found.
- An approach that automates the grouping of multi-part operation clusters into setups. The approach should take into account the similarity between single part clusters and the production cycle time requirements.
- Development of an automated methodology that maps setups to alternative RMTs based on their manufacturing requirements.

CONCLUSION

In this paper, we have presented a model for concurrent process planning and machine selection as a key component in system level process planning for RmS. We have also introduced a new patterning algorithm for parallelism-based operation clustering. The algorithm forms identical operation clusters by first searching for translational vectors and then extracting the end-points. This algorithm is of polynomial computational complexity and is applicable to cases where only translational relations exist between different clusters. The validity of this algorithm is examined through an application example. Such a tool when extended to the general case of rotational/translational patterns and multi-part patterns will be extremely useful for the design of a RmS.

ACKNOWLEDGMENTS

The authors are pleased to acknowledge the financial support of the National Science Foundation (Grant # EEC-9529125) for the Engineering Research Center for Reconfigurable Machining Systems at The University of Michigan at Ann Arbor.

REFERENCES

1. Boerma, J. R. and Kals. H. J. J., "FIXES, a system for automatic selectionof set-ups and design of fixtures", Annals of the CIRP, Vol. 27, No. 1, pp. 443-446, 1988
2. Chu, Chi-Cheng Peter and Gadh, Rajit, "Feature-based approach for set-up minimization of process design from product design", Computer-Aided Design, Vol. 28, No. 5, pp. 321-332, 1996
3. Delbressine, F. L. M., Groot, R. de, and Wolf, A. C. H. van der, "On the automatic generation of set-ups given a feature-based design representation", Annals of the CIRP, Vol. 42, No. 1.,pp. 527-530, 1993
4. Demey, S., Van Brussel, H., and Derache, H., "Determining Set-ups for Mechanical Workpieces", Robotics and computer-integrated manufacturing, Vol. 12, No. 2, pp. 195-203, 1996
5. Han, C. and Ham, I., "Multiobjective Cluster Analysis for Part Family Formations," Journal of Manufacturing Systems, Vol. 5, 223-230, 1985
6. Karp, R. "Reducibility Among Combinatorial Problem", Complexity of Computer Computations, 1972
7. Koren, Yoram and Ulsoy, A. G. "Reconfigurable Manufacturing Systems", ERC/RMS Report 1, September 1997
8. Koren, Y., "Reconfigurable Machining Systems: Vision with Examples", ERC/RMS Report 19, January, 1999
9. Kusiak, A, "The Generalized Group Technology Problem", International Journal of Production Research, Vol. 25, 561-569, 1987
10. Ling, C., Son, S. -Y., Olsen, T., and Yip-Hoi, D., "A System Level Process Planner for Reconfigurable Machining Systems", ERC/RMS Report 24, February 1999
11. Ozturk, F., Kaya, N., Alankus, B., and Sevinc, S., "Machining features and algorithms for set-up planning and fixture design", Computer Integrated Manufacturing System, Vol. 9, No. 4, p. 207-216, 1996
12. Sarker, Bhaba, R., "The Resemblance Coefficients in Group Technology: A Survey and Comparative Study of Relational Metrics", Computer and Industrial Engineering Vol. 30, No. 1, 103-116, 1996
13. Selim, H. M., Askin, R. G., and Varkharia, A. H., "Cell Formation in Group Technology: Review, Evaluation, and Directions for Future Research", Computers and Industrial Engineering, Vol. 34, No. 1, 3-20, 1998
14. Sormaz, D.N., and Khoshnevis, B., "Process Sequencing and Process Clustering in Process Planning Using State Space Search"., Journal of Intelligent Manufacturing, pp. 189 – 200, July, 1996
15. Yut, G. and Chang, T.C., "Heuristic Grouping Algorithm for Fixture and Tool Setups", Engineering Design and Automation, Vol.1, 21-31, 1995
16. Zhang, H. C., and Lin, E. H., "A Hybrid-graph Approach for Automated Setup Planning in CAPP", Robotics and Computer-Integrated Manufacturing, Vol. 15, 89-100. 1999