

Tool-path Optimization for Minimizing Airtime during Machining

Kenneth Castelino¹ Roshan D'Souza¹ and Paul K. Wright¹

1. Mechanical Engineering Dept.
University of California at Berkeley
Berkeley, California 94720, USA

Abstract

This paper describes an algorithm for minimizing the non-productive time or 'airtime' for a tool by optimally connecting the toolpaths for that tool. This problem is formulated as a generalized traveling salesman problem with precedence constraints and is solved using a heuristic method. The performance of the heuristic algorithm and the amount of improvement obtained for different problem sizes is also presented. This algorithm has been implemented in our automated process planning system and can be applied easily to other areas of path planning optimization like fused deposition modeling and laser cutting.

1. Introduction

The total production time for a part consists of two components: the machining time when the tool is actually cutting material and the non-productive time the tool spends in traveling in the air or the 'airtime'. A lot of research has been done on minimizing the machining time during chip generation by optimizing the cutting parameters, the tool sequence [1] and the type of toolpaths to be used. However, less attention has been paid to reducing the non-productive airtime though it too can be quite significant. This is especially true when multiple tools are used for machining since the smaller tools often have to remove a number of small regions and the airtime becomes important. In order to minimize the airtime for a tool, the toolpath regions of the machining features being machined by that tool need to be connected with minimum length. This problem results in a variant of the traveling salesman problem (TSP) but is more complicated since the toolpath regions are contours rather than points and there are precedence constraints due to the way the features are machined.

A model that accurately represents the toolpath connection problem as a generalized traveling salesman problem with precedence constraints has been developed in this research. The model works directly on contours and incorporates constraints easily, which results in an efficient algorithm. This paper focuses on the formulation of the model and describes the heuristic methods that are used to obtain the optimal or near optimal solution. The layout of the paper is as follows: Sec. 2 reviews previous work on similar problems while Sec. 3 gives a brief overview of

the traveling salesman problem and its variants. Sec. 4, 5 describe the formulation of the problem and the heuristic algorithm for solving it. Sec. 6, 7 discuss the performance and scalability of the algorithm and show some sample parts on which the algorithm has been applied.

2. Previous Approaches

Most early attempts at reducing non-productive time used drilling or hole-making processes since they involve only holes, which can be treated as points and are much easier to model. Kolahan et al. [2] used a tabu-search approach to minimize the total processing cost for hole-making operations. They considered tool travel time, tool switching time and the cutting time and used the tabu-search algorithm to find the solution. Manber and Israni [3] considered the problem of minimizing the number of piercing points required in the laser cutting process. In both cases, the problems were modeled as traveling salesman problems with appropriate weights and cost functions but are restricted to single points. The problem of path optimization between polygons gets more complicated because each polygon has multiple vertices, each of which is a possible entry/exit point. Khan et al. [4] considered path optimization between polygons using entry and exit constraints. They represented the polygons as cells with possible entry and exit points and used simulated annealing to solve the problem. Han and Na [5] also used the concept of cells to develop an algorithm for torch path optimization during laser cutting of nested stocks. In both these methods, a random entry/exit point is chosen for each polygon. The problem now reduces to a standard traveling salesman problem, which is solved to get a solution. The entry/exit points of the cells are now changed and the optimization is carried out all over again. While this iterative procedure is easier to implement since it deals only with standard TSP problems, it does not give any guarantees about the quality and convergence of the solution since it is no longer solving the original problem. In addition, these models cannot incorporate precedence constraints, which are essential in representing any realistic machining problem.

3. Review of the TSP and it's Variants

3.1 Traveling Salesman Problem (TSP)

The traveling salesman problem (TSP) is one of the most widely studied combinatorial optimization problems, which has links with many fields of pure and applied mathematics like graph theory and integer programming [6]. Simply stated, a salesman has to visit N cities, visiting each city exactly once and return back to the starting city with minimum cost.

In graph theory terminology, consider a complete digraph $G = (V, A)$ where $V = \{1, \dots, n\}$ is the vertex set that represents the cities, $A = \{(i, j) : i, j \in V\}$ is the arc set and c_{ij} is the cost associated with arc (i, j) i.e the cost of going from city i to j . If $c_{ij} = c_{ji}$, the TSP is symmetric, otherwise it is asymmetric. The problem is to find the minimum cost cycle that includes every node in the graph exactly once i.e. to find the minimum weight Hamiltonian cycle in G .

The TSP can be formulated as an integer-programming problem by considering variables x_{ij} associated with each arc (i, j) . Here, $x_{ij} = 1$ if the arc (i, j) is in the optimal tour and zero otherwise. The IP formulation in terms of the variables x_{ij} is given below:

Minimize $\sum_{(i,j) \in A} c_{ij} x_{ij}$ subject to:

$$x_{ij} \in \{0, 1\} \quad (1)$$

$$\sum_{j \in V} x_{ij} = 1 \quad i \in V \quad (2)$$

$$\sum_{i \in V} x_{ij} = 1 \quad j \in V \quad (3)$$

$$\sum_{i \in S} \sum_{j \in V/S} x_{ij} \geq 1 \quad S \subset V \text{ and } S \neq \Phi \quad (4)$$

Constraints (2) and (3) ensure that the in and out degree of each vertex is one i.e. each city is visited exactly once. Constraint (4) is the sub-tour elimination constraint and ensures that a complete Hamiltonian cycle is obtained rather than a series of disjoint cycles.

Unfortunately, the TSP is an NP-complete problem [7] and hence it is impractical to find the optimal solution for reasonably large sized problems. It is more practical to use heuristic algorithms that have relatively short running times and often give solutions that differ only slightly from the optimal solution. The symmetric TSP is the most widely studied version and has a number of heuristics, which have been surveyed by Laporte et al. [8] and Johnson [9]. Tour construction methods such as the Christofides algorithm [10] and tour improvement heuristics like the Lin-Kernighan algorithm [11] perform excellently and converge quickly to within 1-2% of the optimal solutions for most problems with less than 1000 cities. For the asymmetric case [12], there are fewer heuristics such as

the Kanellakis-Papadimitrou [13] and Zhang [14] algorithms. These methods are feasible for problems with less than 1000 cities though their performance is not as good as the symmetric TSP heuristics. Other optimization methods like simulated annealing [15], tabu search [16], and genetic algorithms [17] have also been used in combination with local search heuristics but these approaches generally do not perform as well as the specialized heuristics for larger sized problems.

3.2 Generalized Traveling Salesman Problem (GTSP)

The GTSP is a generalization of the TSP. In this case, n cities are grouped into m mutually disjoint districts (clusters), and the traveling salesman has to find the shortest route that visits exactly one city in each district and returns to the starting city. Fig. 1 shows a simple instance of the GTSP and the optimal tour for it. The generalized traveling salesman problem (GTSP) is obviously NP-complete, since it reduces to the TSP when each district has exactly one city.

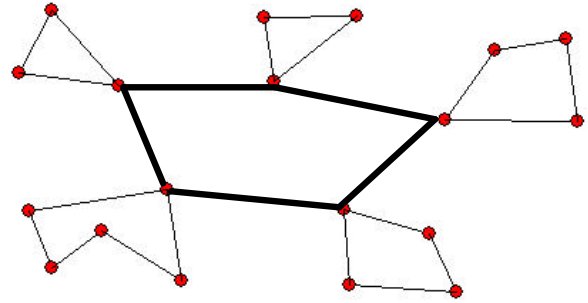


Fig. 1. An instance of the GTSP and its optimal tour.

Laporte and Nobert [23] used a branch and cut algorithm while Renaud [24] proposed a composite heuristic to solve the GTSP directly. Lien [25] and Dimitrijevic [26] used a different approach where they converted a GTSP into a larger size standard TSP such that the optimal solution of this TSP was simply related to the optimal solution of the GTSP. The advantage of this method is that there already exist very efficient heuristics for the TSP, which can be used to solve the problem faster even though the problem size increases.

3.3 Sequential Ordering Problem (SOP)

The Sequential Ordering Problem (SOP) is an asymmetric traveling salesman problem with additional precedence constraints and is used to model problems like production planning [18] and vehicle routing problems with pick-up and delivery constraints. Given a set of n nodes and distances for each pair of nodes, the problem is to find the minimum cost path from node 1 to node n that visits every node exactly once and satisfies the given precedence constraints.

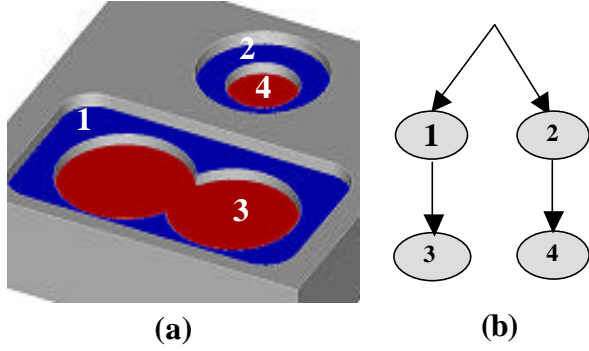


Fig. 2. (a) Sample part and (b) the feature hierarchy.

If the nodes 1 and n are collapsed, then the minimum cost cycle starting and ending at node 1 is obtained, which is the asymmetric TSP. Hence the SOP is essentially an asymmetric TSP with precedence constraints. The generalized version of this problem is also possible and reduces to the generalized TSP with precedence constraints.

Like the TSP, the SOP can also be stated using graph theory. Consider a complete digraph $G = (V, A)$ on n nodes with costs c_{ij} associated with each arc $(i, j) \in A$ and a precedence digraph $P = (V, R)$ that is defined on the same node set V , where an arc $(i, j) \in R$ means that node i must precede node j . The problem now reduces to finding the minimum cost Hamiltonian path in G under the precedence constraints given by P . This problem can be represented as an integer-programming problem and the formulation is similar to that given in Sec 3.1 for the TSP. The only additional equations here are a set of inequalities that enforce the constraints.

The SOP is also NP-complete but it is not as well studied as the TSP and there are few heuristics for solving this problem. Ascheuer et al. [19], [20] developed a branch and cut algorithm as well as a heuristic method for solving the SOP. Chen and Smith [21] gave the first genetic algorithm based approach for the problem while Gambardella [22] used a combination of a genetic algorithm called ant-colony optimization and a local search procedure. This algorithm is the best heuristic method currently available to solve the SOP, since the use of local search greatly improves the quality of the solutions.

4.1 Problem Description

The feature-based planning system of Sundararajan [27] was used in this research. The system takes in a B-Rep model of the part and a tool database and outputs G&M codes for each setup. The number of setups and the machining features to be removed in each setup is determined automatically [28]. Each setup contains a list of features and information about the feature hierarchy. Fig. 2(a) shows a part with a

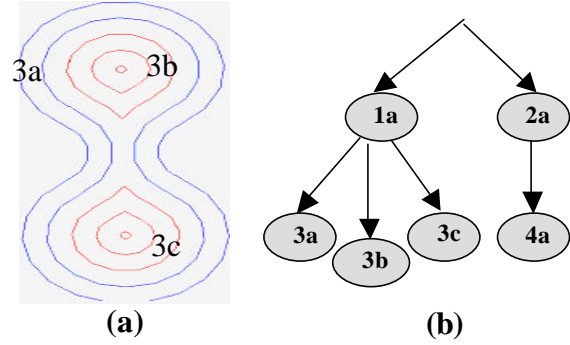


Fig. 3. (a) Toolpath cells (b) Constraints between toolpath cells due to the feature hierarchy..

single setup and Fig. 2(b) shows the hierarchy between features in the setup.

The features in a setup are machined using multiple tools in order to minimize the machining time or cost [1]. The regions of a feature, which are machined using a particular tool, are called the decomposed features for that tool and are machined using contour parallel toolpaths. In order to get the toolpaths for a particular tool, the outer contour of the decomposed feature is successively offset inwards by the tool radius. In general this offsetting process gives a number of concentric toolpath regions since the original contour can split during offsetting. Each of these regions of concentric toolpaths is called a toolpath cell. Fig. 3(a) shows a decomposed feature that splits up into three toolpath cells 3a, 3b, and 3c. So a decomposed feature may have multiple cells and hence a feature can have a number of toolpath cells for a given tool.

A tool starts at the origin, machines all the toolpath cells and finally returns back to the origin, where the next tool replaces it. The tool can move continuously within a toolpath cell without lifting but it has to lift while going from one cell to another resulting in non-productive time. Hence in order to minimize the non-productive time, all the toolpath cells for a tool must be connected with the minimum possible length.

4.2 Problem Formulation

In order to minimize the connecting length, the order in which the cells are visited and the entry/exit point for each cell must be determined. This is like a traveling salesman problem but there are two important differences in this case:

- Due to the feature hierarchy, there is also hierarchy in the toolpath cells i.e. the toolpath cells belonging to a parent feature must be done before those belonging to a child feature. Fig. 3(b) shows the precedence constraints in the toolpath cells due to the feature hierarchy e.g. the toolpath cells of features 3 (3a, 3b, and 3c) must be done after the toolpath cell 1a of feature 1 since feature 3 is a child feature of feature 1.

- The toolpath cells are not points but polygons with a number of possible entry and exit points for each cell. So the entry and exit point for each cell must be found. However, using contour parallel toolpaths simplifies the problem somewhat. Consider the toolpath cell shown in Fig. 4. If the tool enters at point X, it must exit at point Y, otherwise some areas would either remain un-machined or would be over-machined. This means that for a given entry point, the exit point is fixed. So it is sufficient to determine just the entry point for a cell, and the corresponding exit point and the distances to other cells can be computed from this.

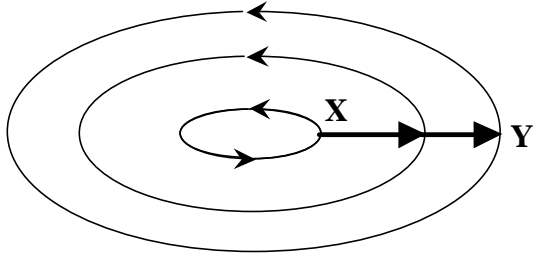


Fig. 4. Given entry point X, the exit point Y is fixed.

Like in Sec 3.2, the cells can be considered as districts and entry, exit points of a cell as the cities in that district. The problem then becomes a generalized traveling salesman problem with precedence constraints, where the tool has to visit exactly one city (entry point) in each district (cell) and return to the starting point (origin) while obeying the precedence constraints among the cells. This model is the most general formulation of the problem and represents the physical process accurately.

5. Solution Algorithm

The algorithm for solving the above problem consists of two steps: the generalized TSP is first converted into a standard TSP while preserving the constraints and this new TSP is then solved using a heuristic algorithm. The solution of the original GTSP

is obtained from the TSP solution since there is a simple relation between the two solutions. The steps are explained in the following sub-sections.

5.1 GTSP to TSP Conversion

The first step converts a GTSP of size n into a standard TSP of size $2n$ [26]. Fig. 5(a) shows a couple of toolpath cells with their candidate entry points and some incoming and outgoing arcs.

The following operations are done on each cell:

- For every node (i) within a cell, make a copy (i') and join them by a arc (i, i'). The arc length of the arc (i, i') is some large number M , where $M > S c_{ij}$. This is the step that changes the size from n to $2n$.
- Connect the original set of nodes by arcs into a cycle. Similarly, connect the new nodes in a cycle but with opposite orientation. All the arcs in these cycles have zero length.
- Arcs that enter the cell remain unchanged but arcs that leave the cell from a particular node (i) are now replaced by arcs from the corresponding new node (i') with the same cost.

These steps are illustrated in Fig 4(b). Consider cell A with nodes 1,2,3. New nodes 1',2',3' are created with arcs $11', 22', 33' = M$. Nodes 1,2,3 and 1',2',3' are connected in cycles with opposite sense with arcs of zero cost. Finally, the original arc (1,6) is replaced by the arc (1',6) of same cost c_{16} .

After these steps, a TSP of size $2n$ is obtained and the solution of this TSP is directly related to that of the original GTSP. If there is a precedence constraint between the 2 cells, the first two steps still need to be done and only the final step changes; e.g. if cell B must be done before cell A, then the cost of going from any node (i) in A to a node (j) in B, c_{ij} is set to -1 . Hence in the figure the cost c_{16} would be set to -1 . This negative weight is used to indicate a constraint to the SOP solver.

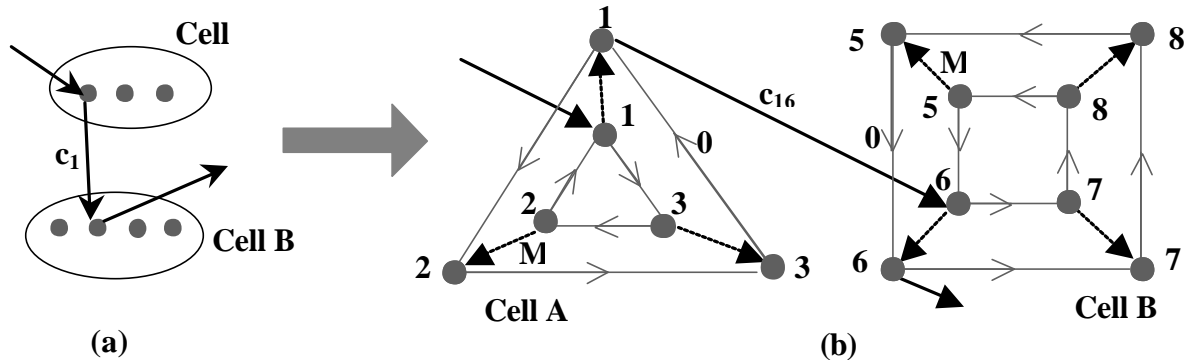


Fig. 5. (a) Two cells with candidate entry/exit points

(b) TSP of size $2n$ obtained from the GTSP

5.2 Solving the TSP

If the original GTSP has constraints, then the new TSP also has constraints and hence a SOP heuristic must be used. However, some parts do not have constraints between features i.e they have a flat feature hierarchy. In such cases, the TSP does not have any constraints and a simple TSP solver can be used. The SOP heuristic of Ascheuer [20] was used for problems with constraints, while the LK heuristic of Helsgaun [29] for problems without any constraints.

The solution of the TSP or the SOP gives both, the order in which the toolpath cells are visited and also the entry point for each cell. Once this information is known, the planner links all the cells in the correct order and generates the final connected toolpath for the tool. This procedure is repeated for each tool in a setup.

6 Algorithm Performance

The standard TSP and SOP instances from the TSPLIB library [30] were used to test the performance of the heuristics. All the tests were carried out on a PC with a 1200 MHz Athlon processor and 512 MB of RAM. Fig. 6 shows the performance of the two algorithms on different sized problems. Both algorithms were able to find the optimal solution for all problems with less than 200 cities and the LK heuristic was within 1% of the optimal solution for problems with less than 1000 cities. The SOP heuristic is not as fast as the LK heuristic but this is understandable since it has to handle additional precedence constraints.

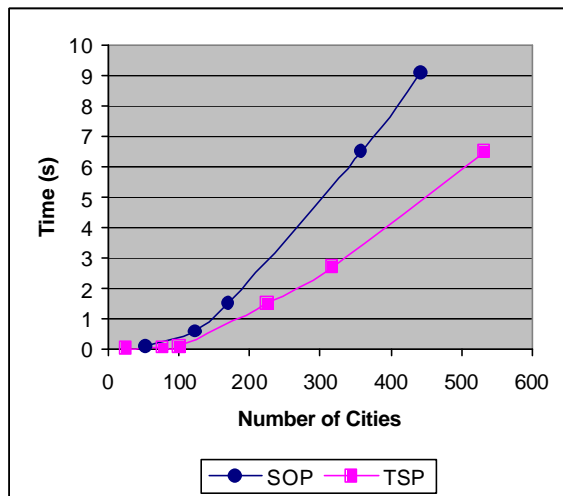


Fig. 6. Performance of the SOP and TSP heuristics

The improvement in the solution was measured with respect to an initial random solution. Fig. 7 shows a problem with 82 polygons and 242 points. The algorithm results in a 544% improvement over an initial random solution. If a solution got from a simple greedy approach is used as the base for comparison, the

improvements are on the order of 65-70%, which is still very substantial. Fig. 8 shows the percentage improvements for problems of different sizes. As the number of cities increases, the improvement in solution increases, which is to be expected.

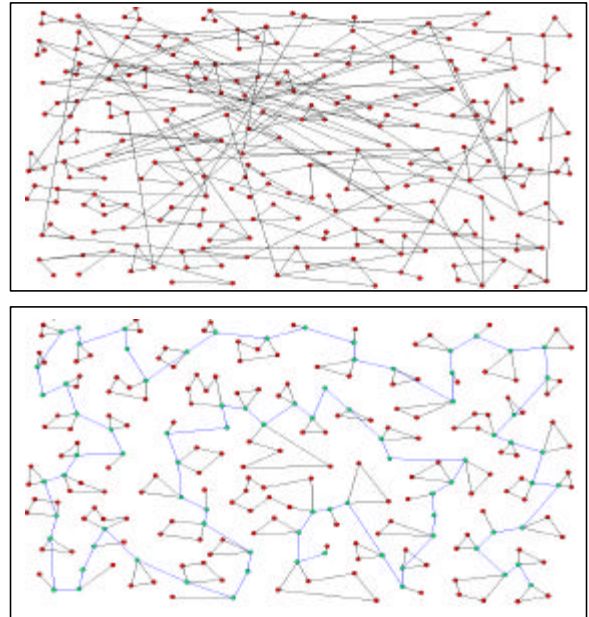


Fig. 7. Random and optimal tours for 242 city problem

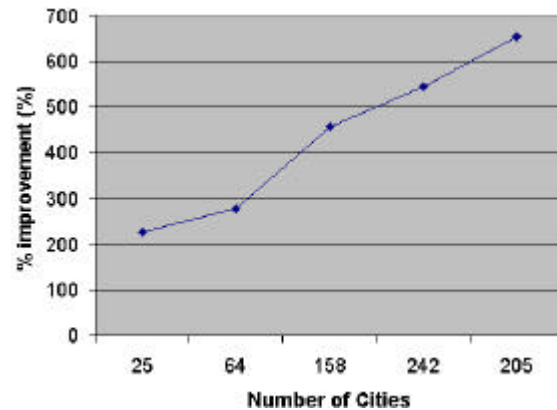


Fig. 8. Tour improvements achieved by the algorithm.

7. Examples

The SOP and TSP algorithms have been incorporated into our process planning system. Fig. 9 shows toolpaths for two parts, which do not have any precedence constraints. The LK heuristic was used to find the solution in these cases. Fig. 10 shows the toolpath for a part with a feature hierarchy, for which the SOP heuristic was used. It can be seen that the cells are connected up optimally while obeying the precedence constraints.

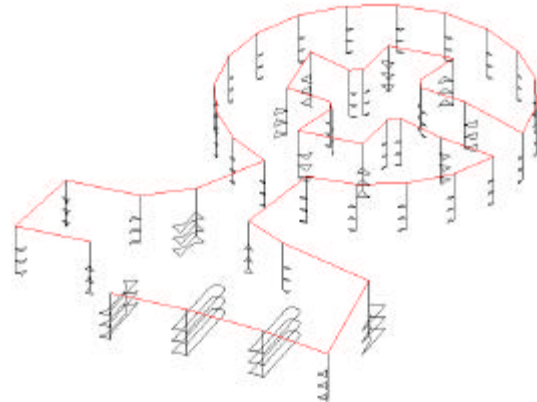
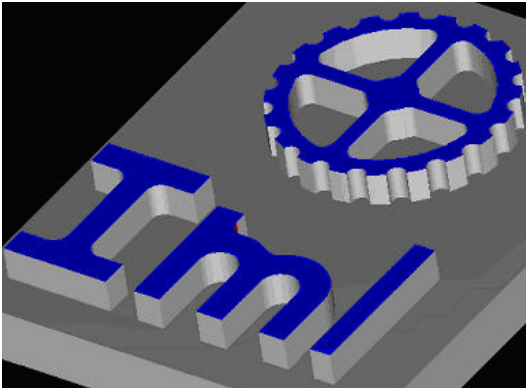
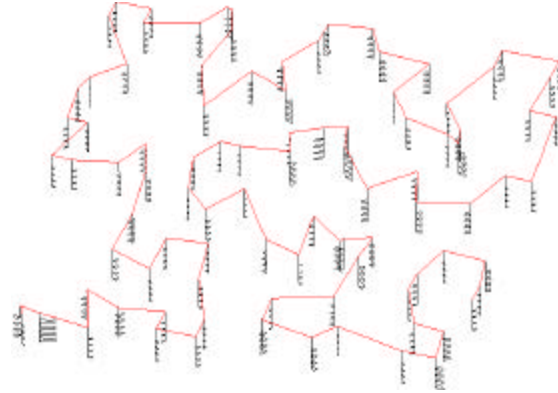
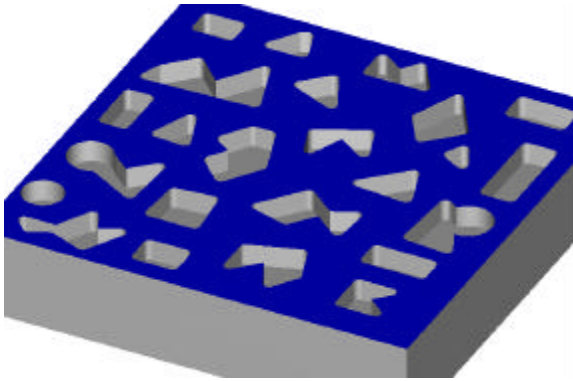


Fig 9. Parts without precedence constraints.

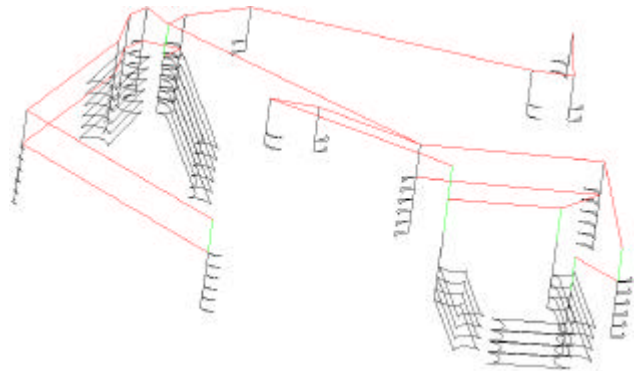


Fig 10. Part with precedence constraints.

8. Conclusions

- The airtime minimization problem was formulated as a generalized traveling salesman problem with precedence constraints. This generalized problem was first converted to a standard problem, which was then solved using either the TSP heuristic or the SOP heuristic depending on whether there were precedence constraints or not.
- The performance and scaling of the heuristic algorithms for the TSP and SOP were studied. Both heuristics gave the optimal solutions for most parts

used in the process planning system and are a viable option for the problem sizes normally encountered.

- The percentage improvement from a starting tour depended on the problem size and as the number of cities increases, the benefit of using this algorithm became clear.
- The basic algorithm is not just restricted to machining but can be applied to path optimizations involving polygons or constraints like in laser cutting or fused deposition modeling.

Acknowledgements

We thank Norbert Ascheuer for providing us with the heuristic algorithm for solving the SOP and Keld Helsgaun for the source code of the Lin-Kernighan heuristic. This work is in collaboration with Prof. Jami Shah at Arizona State University and we thank the Design, Manufacturing, and Industrial Innovation Division of NSF for support.

References

- [1] D'Souza, R., Wright, P. K., and Sequin, C., "Automated Microplanning for 2.5 D Pocket Machining", *Journal of Manufacturing Systems*, 2001.
- [2] Kolahan, F., and Liang, M., "Optimization of hole-making operations: a tabu-search approach", *International Journal of Machine Tools & Manufacture*, 40, 2000, pp. 1735-1753.
- [3] Manber, U., and Israni, S., "Pierce point minimization and optimal torch path determination in flame cutting", *Journal of Manufacturing Systems*, 3(1), 1984, pp. 81-89.
- [4] Khan, W.A., Hayhurst, D.R., and Cannings, C., "Determination of optimal path under approach and exit constraints", *European Journal of Operational Research*, 117, 1999, pp. 310-325.
- [5] Han, G. and Na, S., "A study on torch path planning in laser cutting process Part 2: Cutting path optimization using simulated annealing", *Journal of Manufacturing Process*, 1(1), 1999, pp. 62-70.
- [6] Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G., and Shmoys D.B., "The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization", Wiley, New York, 1985.
- [7] Garey, M.R., and Johnson, D.S., "Computers and Intractability: A guide to the theory of NP-completeness", W.H. Freeman, 1979.
- [8] Laporte, G., "The traveling salesman problem: An overview of exact and approximate algorithms", *European Journal of Operational Research*, 59, 1992, pp. 231-247.
- [9] Johnson, D., McGeoch, L., "The traveling salesman problem: A case study in local optimization", *Local Search in Combinatorial Optimization*, 1997, pp.215-310.
- [10] Christofides, N., "Worst-case analysis of a new heuristic for the travelling salesman problem" Report No. 388, GSIA, Carnegie-Mellon University, 1976.
- [11] Lin, S., and Kernighan, B.W., "An effective heuristic algorithm for the traveling salesman problem", *Operations Research*, 21, 1973, pp. 498-516.
- [12] Cirasella, J., Johnson, D., and et al., "The Asymmetric Traveling Salesman Problem: Algorithms, Instance Generators, and Tests", *ALNEX 2001 Proc.*, Springer Lecture Notes in Computer Science 2153, 2001, pp. 32-59.
- [13] Kanellakis, P., and Papadimitriou, C., "Local search for the asymmetric traveling salesman problem", *Operations Research*, 28(5), 1980, pp. 1066-1099.
- [14] Zhang, W., "Truncated branch and bound: a case study on the asymmetric TSP", *Proc. Of AAAI 1993 Spring Symposium on AI and NP-hard problems*, pp. 160-166.
- [15] Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P., "Optimization by simulated annealing", *Science*, 220, 1983, pp. 671-680.
- [16] Glover, F., "Multilevel tabu-search and embedded search neighborhoods for the traveling salesman problem," Manuscript, School of Business, University of Colorado, Boulder, CO, May 1991.
- [17] Muhlenbein, H., Gorges-Schleuter, M. and Kramer, O., "Evolution algorithms in combinatorial optimization", *Parallel Comput.*, 7, 1988, 65-85.
- [18] Escudero, L.F., Guignard M., and Malik, K., "A Lagrangian relax-and-cut approach for the sequential ordering problem with precedence constraints", *Annals of Operations Research*, 50, 1994, pp. 219-237.
- [19] Ascheuer, N., Jünger, M., and Reinelt, G., A branch & cut algorithm for the asymmetric traveling salesman problem with precedence constraints, *Computational Optimization and Applications*, 17, 2000, pp. 61-84.
- [20] Ascheuer, N., Jünger, M., and Reinelt, G., Heuristic algorithms for the asymmetric traveling salesman problem with precedence constraints – A computational comparison, *Technical Report, ZIB Berlin*, 1998.
- [21] Chen, S. and Smith, S., "Commonality and genetic algorithm", *Technical Report CMU-RI-TR-96-27*, Carnegie Mellon University, Pittsburgh, 1996.
- [22] Gambardella, L.M. and Dorigo, M., "An ant colony system hybridized with a new local search for the Sequential Ordering Problem", *INFORMS Journal on Computing*, 12(3), 2000, pp. 237-255.
- [23] Laporte, G. and Nobert, Y., "Generalized traveling salesman problem through n sets of nodes: An integer Programming approach", *Inform.*vol. 21(1), 1983.
- [24] Renaud, J. and Boctor, F.F., "An efficient composite heuristic for the symmetric generalized traveling salesman problem", *European Journal of Operational Research*, 108, 1998, pp. 571-584.
- [25] Lien, Y.N. and Ma, E., "Transformation of the generalized traveling salesman problem into the standard traveling salesman problem", *Information Sciences*, 74, 1993, pp. 177-189.
- [26] Dimitrijevic, V. and Saric, Z., "An effective transformation of the generalized traveling salesman problem into the traveling salesman problem on digraphs", *Information Sciences*, vol. 102, 1997, pp. 105-110.
- [27] Sundararajan, V. and Wright, P.K., "Identification of Multiple Feature Representations by Volume Decomposition for 2.5D Components", presented at the Winter Annual Meeting of ASME 1998, November 16 - 20, Anaheim, California.
- [28] Kannan, B. and Wright, P. K., "Efficient algorithms for automated process planning of 2.5D machined parts considering fixturing constraints", communicated to the *Intl. J. of Computer Integrated Manufacturing*, 2001.
- [29] Helsgaun, K., "An effective implementation of the Lin-Kernighan traveling salesman heuristic", *European Journal of Operational Research*, 126, 2000, pp. 106-130.
- [30] Reinelt, G., "TSPLIB---a traveling salesman problem library", *ORSA Journal on Computing*, 3, 1991, pp. 376-384.