A Geometric Algorithm for Selecting Optimal Set of Cutters for Multi-Part Milling

Zhiyang Yao
Mechanical Engineering Department
and Institute for Systems Research
University of Maryland
College Park,MD-20742
+1-301-405-6572
yaodan@Glue.umd.edu

Satyandra K. Gupta
Mechanical Engineering Department
and Institute for Systems Research
University of Maryland
College Park,MD-20742
+1-301-405-5306
skgupta@eng.umd.edu

Dana S. Nau
Computer Science Department
and Institute for Systems Research
University of Maryland
College Park,MD-20742
+1-301-405-2684
nau@cs.umd.edu

ABSTRACT

For the manufacture of milled parts, it is well known that the size of the cutter significantly affects the machining time. However, for small-batch manufacturing, the time spent on loading tools into the tool magazine and establishing *z*-length compensation values is just as important. If we can select a set of milling tools that will produce good machining time on more than one type of parts, then several unnecessary machine-tool reconfiguration operations can be eliminated. This paper describes a geometric algorithm for finding an optimal set of cutters for machining a set of 2½D parts. In selecting milling cutters we consider both the tool loading time and the machining time and generate solutions that allow us to minimize the total machining time. Our problem formulation addresses the general problem of how to cover a target region to be milled with a cylindrical cutter without intersecting with the obstruction region; this definition allows us to handle both open and closed edges in the target region. Our algorithm improves upon previous work in the tool selection area in following ways: (1) in selecting cutters, it accounts for the tool loading time, and (2) it can simultaneously consider multiple different parts and select the optimal set of cutters to minimize the total manufacturing time.

Keywords

Milling Cutter Selection, Computer-Aided Process Planning.

1. INTRODUCTION

Increasingly, the manufacturing industry is moving towards high part mixes, which makes it important to reduce setup and tooling operations. For example, if a machine-tool is not configured to accommodate more than one part within a part family, then large amounts of time will repeatedly be spent on reconfiguring the machine-tool (i.e., loading new tools and fixtures into the machine-tool) each time a request is received for manufacturing a different part. Such reconfigurations are the major source of inefficiency in small batch manufacturing.

If a machine-tool was configured from the beginning to accommodate several different parts within the part family, much of the cost of reconfiguring the machine-tool could be avoided. This will require considering all of the parts that need to be produced during the given operational period, and selecting tools and machine-tool configurations that can work for multiple different parts.

In the milling operation domain, it is well known that the size of the milling cutters significantly affects the machining time. Therefore, in order to perform milling operations efficiently, we need to select a set of milling cutters with optimal sizes. It is difficult for human process planners to select the optimal or near optimal set of milling cutters due to complex geometric interactions among tools size, part shapes, and tool trajectories. Furthermore, in small batch manufacturing, both tool loading time (i.e., the time spent on loading tools into the tool changer) and machining time (i.e., the time spent on performing milling operations) are equally important.

Most existing cutter selection algorithms select milling cutters by minimizing the machining time and do not account for tool loading time [1, 2, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]. In most cases, the existing algorithms will recommend using a different set of cutters for each new type of part. Since most machine-tools can only hold a limited number of tools at one time, this means that we will need to reconfigure the machine-tool (i.e., we will need to change the set of tools in the tool magazine) before machining each new type of part. When the batch size is small, reconfiguring the machine-tool before machining each type of part may significantly reduce the throughput. However, if we can select a set of tools that can be used for more than one type of part, then several unnecessary machine-tool reconfiguration operations can be eliminated, thereby increasing the throughput.

This paper describes a geometric algorithm for finding an optimal set of milling cutters for machining a given set of parts. In selecting milling cutters we consider both the tool loading time as well as machining time and generate solutions that allow us to minimize the total manufacturing time. Our tool selection algorithm improves upon the previous work in this area, in the following manner: (1) in selecting cutters it accounts for tool loading time, and (2) it can simultaneously consider multiple different parts and select the optimal set of cutters to minimize the total manufacturing time.

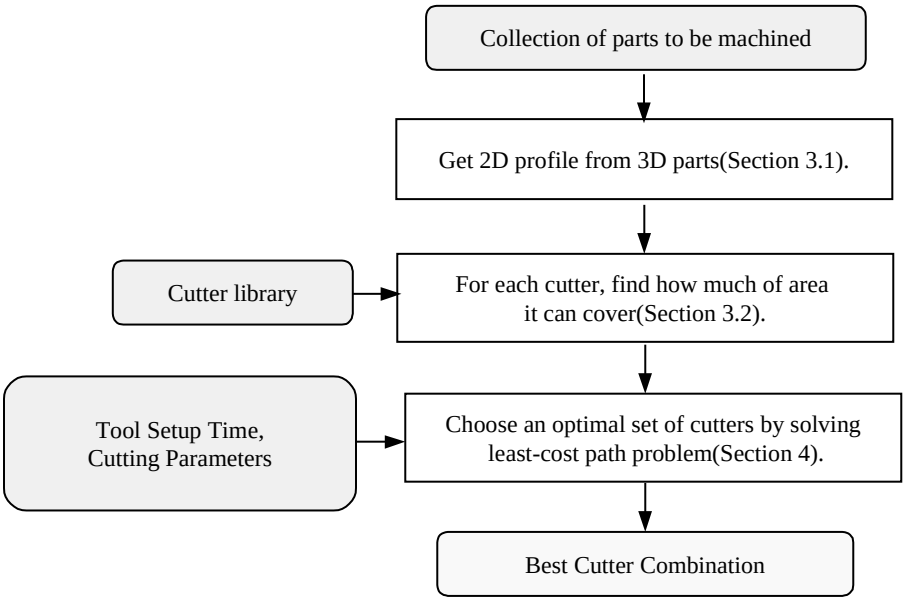


Figure 1: Overview of Our Approach

Currently our algorithm is restricted to 2½D milling operations. In particular, we consider the problem of selecting a sequence of cylindrical cutters to cut all of the points in a 2½D target region without cutting any of the points in a 2½D obstruction region [15]. This approach allows us to handle both open and closed edges. The steps of our approach are as follows. Given a set of available cutters, we first compute how much of the target region each cutter can cut. We do this by finding the set of all possible permissible locations for the tool, and then computing the total area covered by the tool at these permissible locations.[1] Next, we represent the problem of finding an optimal sequence of cutters as a least-cost path problem, and use Dijkstra's algorithm to solve it. Our overall approach is shown in Figure 1.

2. PROBLEM FORMULATION

The milling problem is the problem of taking one or more pieces of stock and using a sequence of one or more milling operations to remove portions of each piece of stock, in order to produce some desired set of parts. Each milling operation is performed using a milling cutter, and our research focuses on the geometric aspects of selecting those cutters. In previous work [15], we looked at the case where only one milling operation was to be used, and developed an algorithm for finding the optimal cutter for this operation. However, in practical milling problems, it is more typical to use more than one milling operation, using a different cutter for each operation—and that problem is the subject of the current paper.

Let P be one of the parts that needs to be produced, and let S be the piece of stock from which P is to be produced. We will assume that S –* P (i.e., the portion of S that needs to be removed to produce P) is a 2½D solid (this assumption holds for many milling operations). In this case, the cutter selection problem can be reduced to a 2D problem by considering any cross-section of the 2½D solid. Within this 2D cross-section, we define the region to be machined as the *target region T* (a region is a set of 2D points). In addition to the target region, there is an *obstruction region O* that is the region that the cutting tool should not cut during machining. An example is shown in Figure 2. The target region and the obstruction region must both be regular sets, each of which may consist of a number of non-adjacent sub-regions:

$$T = T_1 \cup \ldots \cup T_M;$$

$$O = O_1 \cup \ldots \cup O_N.$$

In this paper, we assume that the boundary of each sub-region consists only of line segments and segments of circles.

Let C be a rotating cutter of radius $r(C)$ located at some point $p=(x,y)$. If we hold C stationary while it is rotating, then C will cut a circular region $R(C,p)$ = {all points (u,v) such that $\sqrt{(u-x)^2+(v-y)^2} \le r(C)$ }. We will call $R(C,p)$ the set of points *covered by C at p*.

[1] This problem is computationally similar to the problem of computing the offset for a 2D point set, and previous approaches for this problem have been based on the use of the offsetting operators traditionally available in solid modeling systems. However, in this paper we show that use of traditional offsetting operators will not always produce correct results, and in Section 3.2.2 we show what modifications are needed to make those approaches correct.

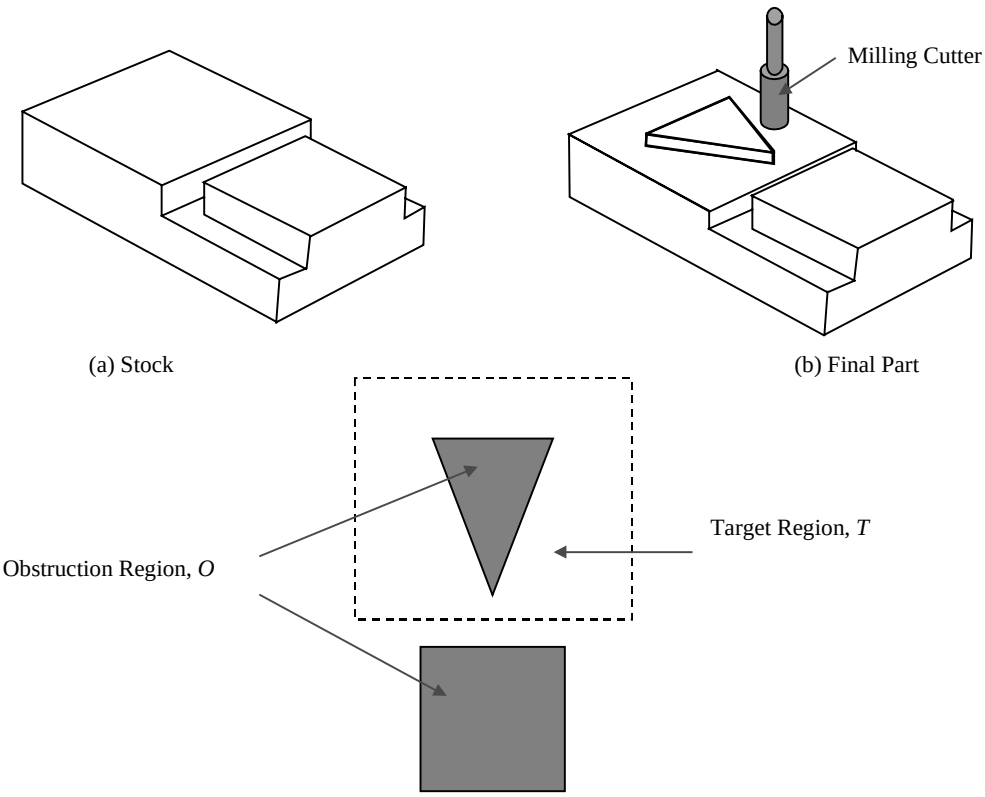


Figure 2: Example et region, and obstruction region.

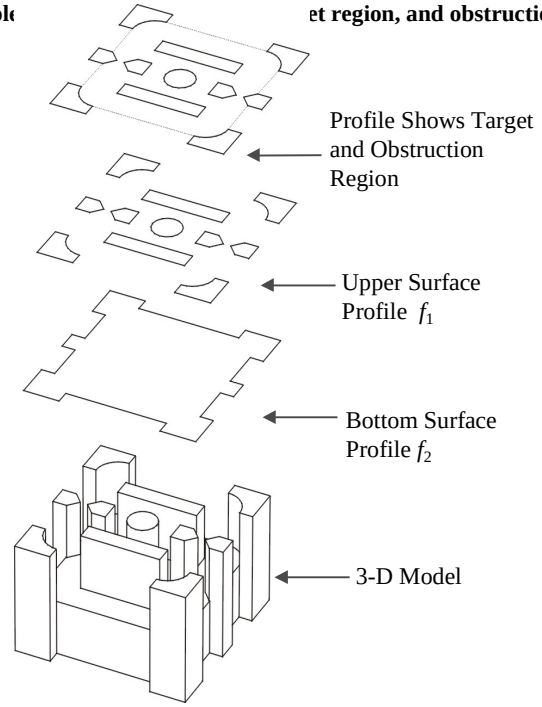


Figure 3: Example of profile extraction

If we move C while it is rotating, then C will cut some region larger than $R(C,p)$. In this paper, one of the things we will want to find is the set of all points in T that can be cut by C. There are several different non-equivalent ways to define what this set is. As we discussed in

[15], the best one for our purposes is as follows. A point p is a *permissible location* for C if the interior of $R(C,p)$ does not intersect with the obstruction region, or equivalently, if $O \cap^* R(C, p) = \emptyset$. A set of points can be *safely covered* by C if for every point p in the set, there is a permissible location of C that covers p.

In most cases, the length of the cutter path associated with a cutter will decrease as the cutter size increases, and therefore it will take less time to cut a given area. However, due to geometric restrictions, a large cutter may not be able to safely cover the entire target region—and thus one or more smaller cutters will be needed to cut the remaining portion of the target region.

In multi-cutter selection problems, multiple milling operations are used, each with a different milling cutter. The bigger cutters are used first, in order to cut material as fast as possible. Then, smaller cutters are used to create the smaller features of the target region. The *total machining time* T_M for the sequence of milling operations is the total time needed from the time that the stock is loaded into the milling machine, until the time that the finished part is produced. T_M can be expressed as

$$T_M = T_{ct} + T_{cc} + T_{cl} ,$$

where T_{ct} is the total real cutting time (the time spend on moving cutters to cut the profile); T_{cc} is the total cutter change time (the total time of changing tools during machining all the parts); and T_{cl} is the total cutter loading time (the total time spent on loading and calibrating all selected cutters before machining given parts). Since cutter change time is significantly smaller (of the order of 5 seconds) compared to cutting time and cutter loading time (of the order of 5 to 10 minutes), in this paper we will ignore cutter change time. Therefore, in this paper we will use

$$T_M = T_{ct} + T_{cl} .$$

Intuitively, the cutter selection problem can be described as a *region covering* problem: we need to find an optimal sequence of cutters that can cover the target region without intersecting the obstruction region. Mathematically, we define the *multi-part cutter selection problem* as follows. Suppose we are given one or more pieces of stock (S_1, …, S_L) from which we need to produce a corresponding set of parts (P_1, …, P_L). In order to produce those parts, suppose we have a sequence of cutting tools (C_1,C_2,…,C_n), given in decreasing order of cutter radius (i.e., $r(C_1) > \ldots > r(C_n)$). Furthermore, suppose that C_n is small enough that it can safely produce all of P_1, …, P_L, and that for $m < n$, no C_m is small enough to safely produce all of P_1, …, P_L. The problem is to find a subsequence (C^*_1,C^*_2,…,C^*_m) of (C_1,C_2,…,C_n) such that if we use C^*_1,C^*_2,…,C^*_m in the order given, this will minimize the total machining time T_M. In this paper we present an algorithm for solving this problem.

3. FINDING COVERABLE AREA FOR A GIVEN CUTTER

In order to solve the multi-part cutter selection problem, an important step is to estimate the area of the region that can be safely covered (in following sections, we call this region as *coverable region* and the area of coverable region as *coverable area*) by each of the cutters C_1, …, C_n. This section describes geometric algorithms for calculating the coverable area for a given part and tool combination.

3.1 An Algorithm for Extracting 2D Profile

To estimate the coverable area automatically, we will need to extract the target and obstruction region from the CAD model. To see how we extract the target region and obstruction region, consider example shown in Figure 3, in which we have a 3D model of a rectangular part whose faces are parallel to the *xy*, *yz*, and *xz* planes. As shown in the figure, this part has a single feature, which is a blind 2½D milling feature which is located in the part's top face. To find the target region and the obstruction region, we extract two cross-sections that are parallel to the *xy* plane: a cross-section f_1 at the top of the part, and a cross-section f_2 at the bottom of the feature. The obstruction region is f_1, and the target region is $f_2 -^* f_1$.

3.2 An Algorithm for Finding Coverable Area

3.2.1 Definitions

Lemma 1: Given a cutter C of radius $r(C)$, the target region T and obstruction region O, the set of non-permissible locations $\boldsymbol{A}(O,C)$ for C is given by:

$$\boldsymbol{A}(O,C) = \{p: \exists\, q \in O, \text{distance}(p,q) < r(C)\}.$$

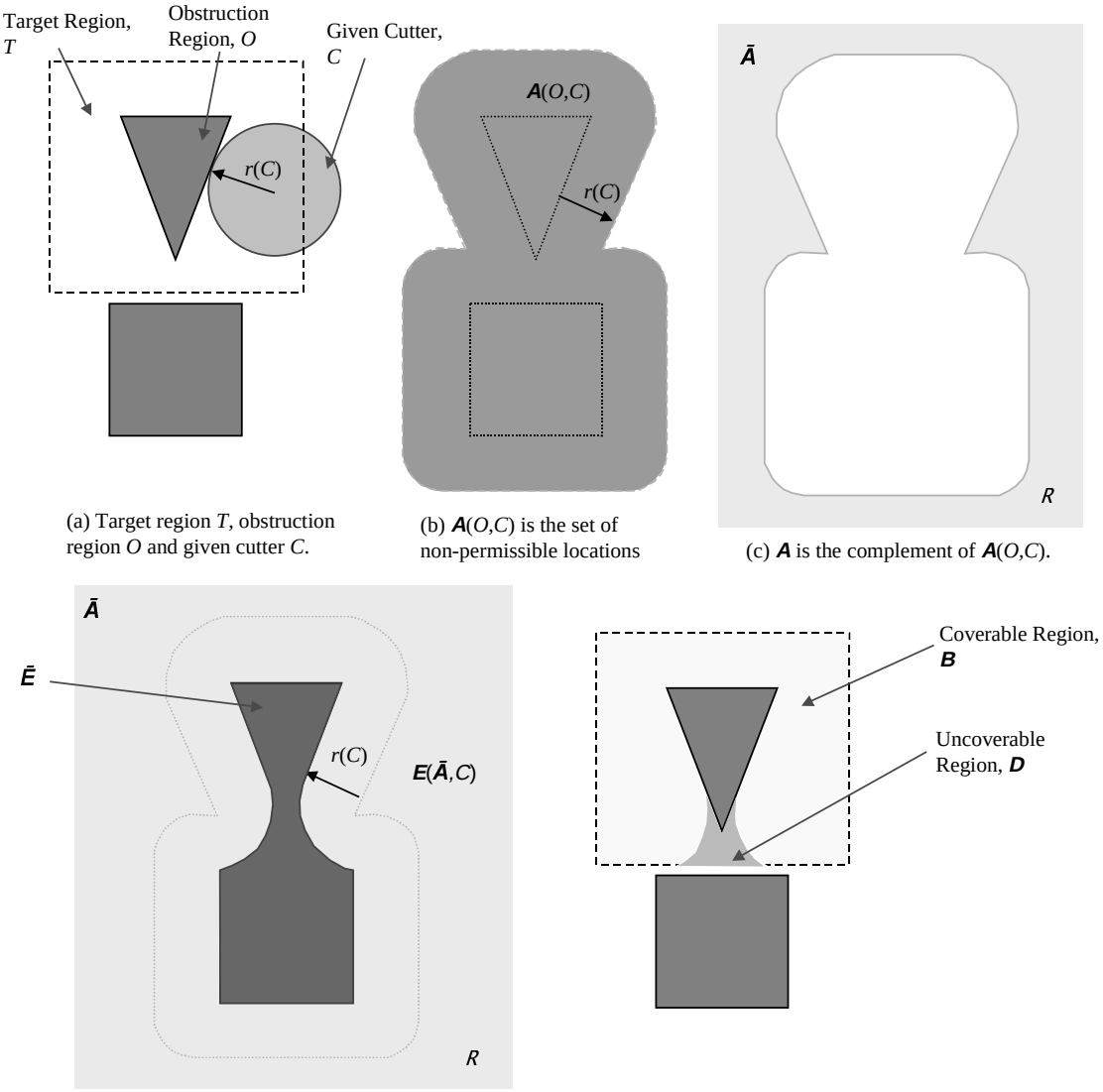
Proof: For every point p in $\boldsymbol{A}(O,C)$, $\exists\, q \in O$ such that distance$(p,q) < r(C)$. Therefore, q will be in the interior of $R(C,p)$. Thus, $R(C,p) \cap^* O \neq \emptyset$. Hence every p in $\boldsymbol{A}(O,r)$ is not a permissible location of C. □

Lemma 2: Let $\bar{\boldsymbol{A}}$ be the complement of $\boldsymbol{A}(O,C)$. Every point in the set $\bar{\boldsymbol{A}}$ is a permissible location.

Proof: . Let p be a point in $\bar{\boldsymbol{A}}$. Let q be the closest point in O to p. Since $\boldsymbol{A}(O,C)$ contains all points whose minimum distance to O is less than $r(C)$, it follows that distance$(p,q) \geq r(C)$. Therefore, q is either outside of $R(C,p)$ or on the boundary of $R(C,p)$. Thus, $R(C,p) \cap^* O = \emptyset$. Hence every p in $\bar{\boldsymbol{A}}$ is a permissible location of C. □

Lemma 3: Let $\boldsymbol{E}(\bar{\boldsymbol{A}},C) = \{p: \exists\, q \in \bar{\boldsymbol{A}}, \text{distance}(p,q) \leq r(C)\}$. Then for every point p in $\boldsymbol{E}(\bar{\boldsymbol{A}},C)$, there is a permissible location q such that p can be safely covered by C at q.

Proof: Let $p \in \boldsymbol{E}(\bar{\boldsymbol{A}},C)$. Then there is a point q in $\bar{\boldsymbol{A}}$ such that distance$(p,q) \leq r(C)$. Therefore, $p \in R(C,q)$. Since q is in $\bar{\boldsymbol{A}}$, it follows from Lemma 2 that q is a permissible location. Therefore, p can be safely covered by q. □



(a) Target region T, obstruction region O and given cutter C.

(b) $\boldsymbol{A}(O,C)$ is the set of non-permissible locations

(c) $\boldsymbol{A}$ is the complement of $\boldsymbol{A}(O,C)$.

(d) $\boldsymbol{E}(\boldsymbol{A},C) = \{p: \exists\, q \in \boldsymbol{A}, \text{distance}(p,q) \le r(C)\}$

(e) Examples of coverable region $\boldsymbol{B}$ and uncoverable region $\boldsymbol{D}$.

Figure 4: Illustration of Definitions

Lemma 4: Let $\bar{\boldsymbol{E}}$ be the complement of $\boldsymbol{E}(\bar{\boldsymbol{A}},C)$. For every $p \in \bar{\boldsymbol{E}}$, there is no permissible location for C to cover p.

Proof: Let p be any point in $\bar{\boldsymbol{E}}$, and suppose we can find a permissible location q such that $R(q,C)$ covers p. Then from Lemma 1, it follows that q is not in $\boldsymbol{A}(O,C)$ and hence is in $\bar{\boldsymbol{A}}$. Since $R(q,C)$ covers p, this means that distance$(q, p) \le r(C)$. Thus p is in $\boldsymbol{E}(\bar{\boldsymbol{A}},C)$, which is a contradiction since p is in $\bar{\boldsymbol{E}}$. □

Theorem 1: Let $\boldsymbol{D} = T -^{*} \boldsymbol{E}(\bar{\boldsymbol{A}},C)$. Then for every $p \in \boldsymbol{D}$, there is no permissible location for C to cover p.

Proof: Since $\boldsymbol{D}$ is a subset of $\bar{\boldsymbol{E}}$, this theorem directly follows from Lemma 4. □

Theorem 2: Let $\boldsymbol{B} = T -^{*} \boldsymbol{D}$. Then for every $p \in \boldsymbol{B}$, there is a permissible location for C to cover p. (Thus we will call $\boldsymbol{B}$ the *coverable region* of C).

Proof: $\boldsymbol{B}$ is a subset of $\boldsymbol{E}$. Therefore, this theorem directly follows from Lemma 3. □

Figure 4 shows an example in which $\boldsymbol{A}(O,C)$, $\bar{\boldsymbol{A}}$, $\boldsymbol{E}(\bar{\boldsymbol{A}},C)$, $\bar{\boldsymbol{E}}$, $\boldsymbol{D}$ and $\boldsymbol{B}$ are given.

3.2.2 Algorithm

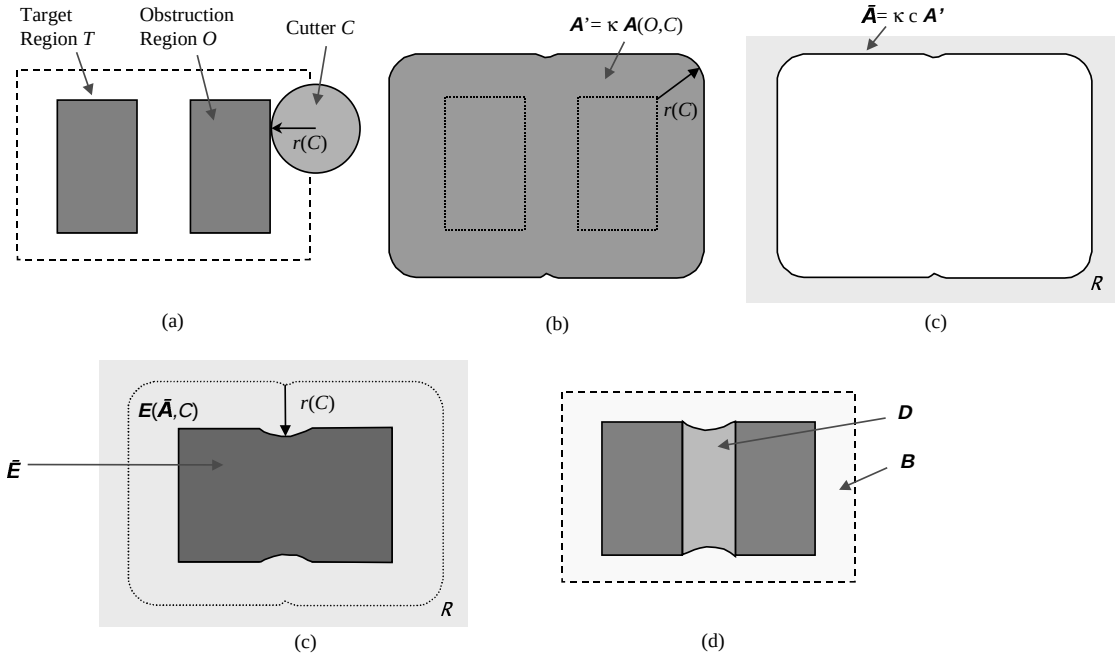


Figure 5: A case where using the offset operator to compute $\boldsymbol{A}'$ results in a correct value for $\boldsymbol{B}$.

Let A be the area of the coverable region $\boldsymbol{B}$. A is the coverable area of using C. From the definition and lemmas introduced in Section 3.2.1, we know that if we can find $\boldsymbol{D}$, then we can get $\boldsymbol{B}$ and then we can compute A. Our algorithm for doing this is as follows:

COVERABLE_AREA_FINDING(C, O, T)

1. From O and $r(C)$, get the 2D point set $\boldsymbol{A}(O,C)$ (see the discussion of this below)
2. $\bar{\boldsymbol{A}}$=c$\boldsymbol{A}(O,C)$ (c is the operator to compute complement, and we assume that the universal set is a large rectangular area that contains $\boldsymbol{A}$)
3. From $\bar{\boldsymbol{A}}$ and C, get $\boldsymbol{E}(\bar{\boldsymbol{A}},C)$
4. $\bar{\boldsymbol{E}}$=c$\boldsymbol{E}$

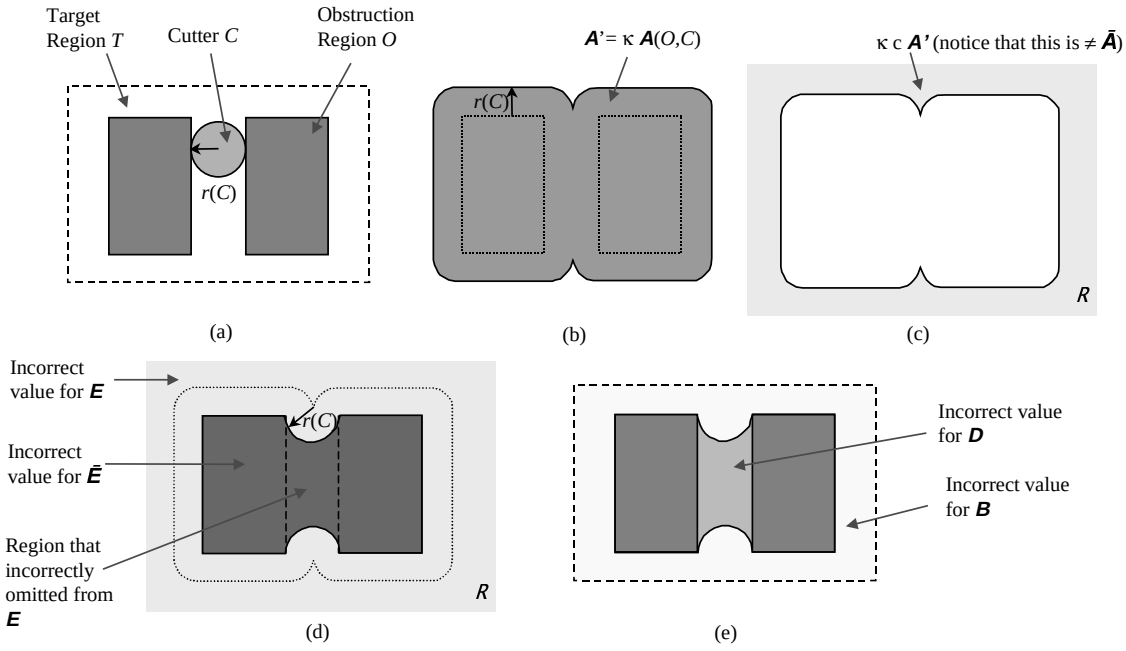


Figure 6: A case where using the offset operator to compute $\boldsymbol{A}'$ results in an incorrect value for $\boldsymbol{B}$.

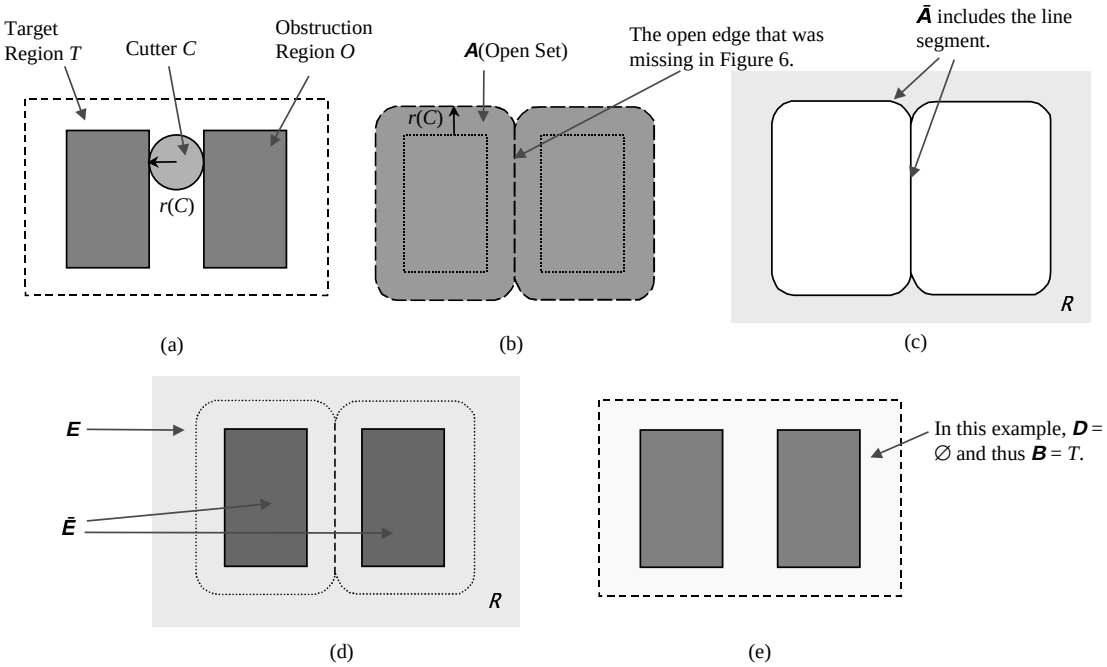


Figure 7: A correct way to compute $\boldsymbol{B}$ for part shown in Figure 6

5. $\boldsymbol{D} = T -^{*} \boldsymbol{E}(\bar{\boldsymbol{A}},C)$
6. $\boldsymbol{B} = T -^{*} \boldsymbol{D}$
7. Return A = the area of $\boldsymbol{B}$

Except for Step 1, all of the above steps are standard geometric operations, and descriptions of them can be found in geometric modeling books such as [4]. However, Step 1 is more problematic. It involves computing $\boldsymbol{A}(O,C)$, whose definition is similar to the "offset" operation found in popular geometric kernels, but with an important difference. $\boldsymbol{A}(O,C)$ is set of points that *are less than distance r* from the obstruction region. Most offset operators compute the set of points that are *less than or equal to distance r* from the given region. Therefore the set $\boldsymbol{A}(O,C)$ is an open set, whereas the offset operation produces the closed set $\boldsymbol{A'} = \kappa\boldsymbol{A}(O,C)$ (where κ is the closure operator).

In many cases, $\bar{\boldsymbol{A}} = \kappa c\boldsymbol{A'}$. In such cases, Steps 1 and 2 of the COVERABLE_AREA_FINDING algorithm can be replaced with the two steps shown below (for an example, see Figure 5):

1'. compute $\boldsymbol{A'}$ using the offsetting operation

2'. $\bar{\boldsymbol{A}} = \kappa c\boldsymbol{A'}$

In fact, Steps 1' and 2' seem so obvious that they are standardly used in algorithms for computing the area of $\boldsymbol{B}$ [9,10]. However, there are a few cases in which Steps 1' and 2' will lead to incorrect results. For example, suppose that the part is the same as in Figure 5 but the cutter's radius is as shown in Figure 6. In this case, $\bar{\boldsymbol{A}}$ is not equal to $\kappa c\boldsymbol{A'}$, because $\kappa c\boldsymbol{A'}$ does not include the additional edge shown in Figure 7(b) and 7(c). As a result, if we use Steps 1' and 2' instead of 1 and 2, Step 3 will compute an incorrect value for $\boldsymbol{E}$ in that leaves out the "omitted region" shown in Figure 6(d). Thus, if we use Steps 1' and 2' instead of 1 and 2 in the COVERABLE_AREA_FINDING algorithm, we will get an incorrect value for $\boldsymbol{B}$.

In our current implementation of COVERABLE_AREA_FINDING, we use the following approach to compute a close approximation of $\boldsymbol{B}$. We use Steps 1' and 2', but instead of offsetting O by the radius r, we offset O by a distance r', where $r' = r - \varepsilon$ (with $\varepsilon > 0$ and $\varepsilon << r$). This can overcome the problem with the "omitted region" illustrated in Figure 6, but only if the value of ε is chosen carefully so that another "omitted region" does not occur elsewhere.

We are currently extending our implementation to compute set $\boldsymbol{A}(O,C)$ exactly. This requires computing open sets and performing Boolean operations on open sets.

4. FINDING OPTIMAL SEQUENCE OF CUTTERS FOR MULTI-PART

In cutter selection problems, we are given a set of parts associated with corresponding stocks, and a set of available cutters. We need to select a subset of the initial set of cutters such that by using the subset to perform machining operations, the given set of parts can be produced from the corresponding stocks in the shortest possible total machining time.

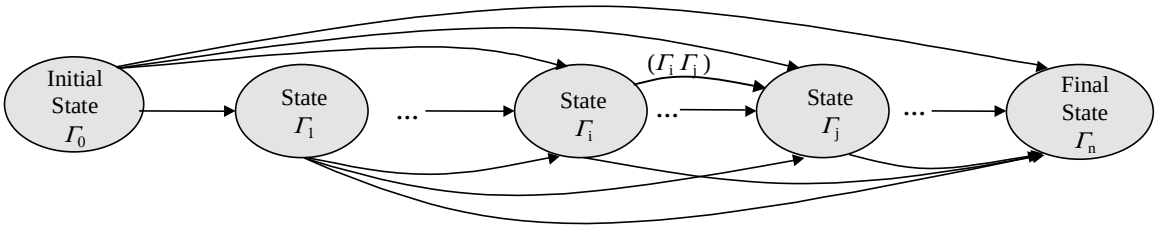


Figure 8: Problem Representation

Recall from Section 2 that we are given a sequence of cutting tools $(C_1,C_2,\ldots,C_n)$, listed in decreasing order of cutter radius; we are given one or more pieces of stock $(S_1, \ldots, S_L)$ from which we need to produce a corresponding set of parts $(P_1, \ldots, P_L)$; and the problem is to find a subsequence $(C^*_1,C^*_2,\ldots,C^*_m)$ of $(C_1,C_2,\ldots,C_n)$ such that if we use $C^*_1,C^*_2,\ldots,C^*_m$ in the order given, we can minimize the total machining time T_M..

We now define the *workpiece state* Γ_{ij} as follows. For j=1,…,L, let $\Gamma_{0j} = S_j$, and for i=1,…,n, let Γ_{ij} be the state of the workpiece that results after using the cutter C_i, under the assumption that we use C_i to cut as much of T_j as it can safely cut. From this definition, it follows that for every i>0, Γ_{ij} is equal to the set of points in T_j that cannot be safely covered by C_i. The reason for this is that any cutters that we used prior to C_i are larger than C_i, and thus the portion of T_j that they can safely cut is a subset of the portion of T_j that C_i can safely cut.

For the given set of parts $(P_1, \ldots, P_L)$, we define the *composite state* Γ_i to be $(\Gamma_{i0}, \Gamma_{i1},\ldots, \Gamma_{iL})$. Thus there are n+1 composite states $\Gamma_0,\ldots,\Gamma_n$. Since C_n can completely cover all of the target regions, Γ_n represents the set of all of the final part shapes.

Let $\boldsymbol{B}_{ij} = T_j -^* \Gamma_{ij}$, and let A_{ij} be the area of $\boldsymbol{B}_{ij}$. (As a special case, note that $\boldsymbol{B}_{0j} = T_j -^* \Gamma_{0j} = T_j -^* S_j = \varnothing$, and thus $A_{0j} = 0$.) Then the safely coverable area for the composite state Γ_i using cutter C_j is given by

$$A_i = \sum_{j=1}^{L} A_{ij} \cdot$$

Let G be the directed graph whose node set is $(\Gamma_0,\ldots,\Gamma_n)$, and whose edge set is $\{(\Gamma_i,\Gamma_j) : i < j\}$.

Each edge (Γ_i,Γ_j) corresponds to the operation of using the cutter C_j to produce Γ_j directly from Γ_i. For each edge (Γ_i,Γ_j), we want to assign a cost $w(\Gamma_i,\Gamma_j)$ that estimates the cost of performing that operation. We will define $w(\Gamma_i,\Gamma_j)$ as follows.

As discussed in Section 2, the cost of performing the operation is $T_{cl} + T_{ct}$, where T_{cl} is the cutter loading time and T_{ct} is the cutting time. An average value for T_{cl} is usually determined experimentally, and we can estimate T_{ct} as follows. Since the cutter C_i has already been used to cut a coverable area A_i of Γ_i, the cutter C_j will only need to cut the area $A_j - A_i$ in order to produce Γ_j. The time needed to cut $A_j - A_i$ can be estimated as $k \times (A_j - A_i)/r_j$, where k is a factor determined by machining parameters. Thus, we define $w(\Gamma_i,\Gamma_j) = T_{cl} + k \times (A_j - A_i)/r_j$.

Since Γ_0 represents the initial stocks $(S_1, \ldots, S_L)$ and Γ_n represents the final parts $(P_1, \ldots, P_L)$, any path in G starting from Γ_0 to Γ_n represents a cutting sequence in which the final parts can be produced form the initial stocks. Because the cost of every edge in G represents the estimated machining time needed to go from one state to another, any valid path in G has an associated total estimated machining time, which is the sum of the path's edge costs. If we can find the least-cost path from Γ_0 to Γ_n, this will give us the sequence of cutting operations that minimizes the total estimated machining time. Using Dijkstra's algorithm, this least-cost path can be found in time $\mathbf{O}(n^2)$ [3].

5. IMPLEMENTATION AND EXAMPLES

We have implemented our algorithm, using C++ and the ACIS® kernel. As an example, Figure 9 shows parts P_1, P_2, P_3 and P_4. In this example, we are given 10 cutters $(C_1,\ldots,C_{10})$ and their radii are 2.5mm, 5 mm, 7.5 mm, 10 mm, 12.5 mm, 15 mm, 17.5 mm, 20 mm, 22.5

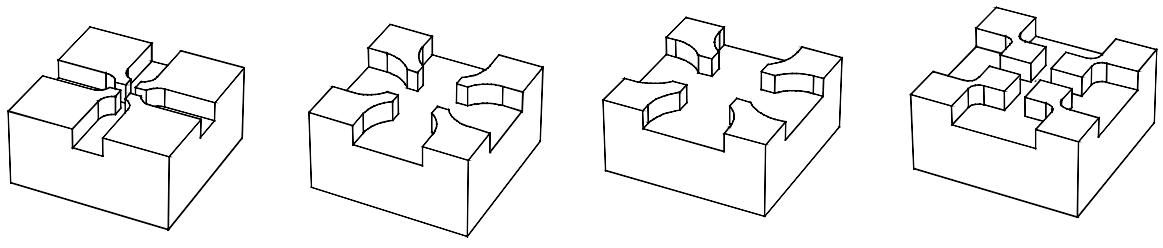


Figure 9 : Example Parts

mm and 25mm. We first get the 2D profile for each part as described in Section 3.1. After that, we use the algorithm described in Section 3.2.2 to get the coverable area for each cutter and part combination. Based on these results, Figure 10 gives a chart showing the relationship between the sizes of the cutters $C_1,\ldots,C_{10}$ and the total area that each cutter can safely cover. After we get all the coverable area values, we use the approach described in Section 4 to find the best combination of cutters. Figure 11 shows the resulting answer.

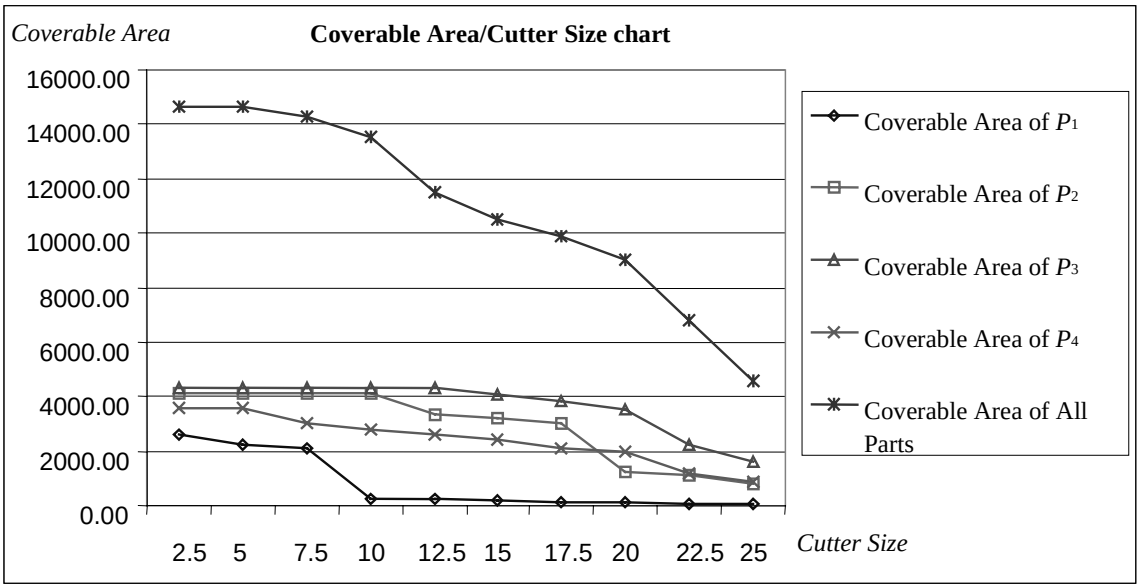


Figure 10 : Cutter Size/ Coverable Area Chart

In Section 1, we pointed out that the best combination of cutters is likely to be different than what we would get by selecting optimal cutter sets for each part individually. As an illustration of this, Figure 12 shows what cutters we would have chosen if we had selected optimal cutter sets for each part individually. If optimal cutter sets are generated for each part individually, then the total number of cutters selected will be 7 (their radii are 2.5 mm, 5 mm, 7.5 mm, 10 mm, 12.5 mm, 17.5 mm and 20 mm). As shown in Figure 13, the total machining time used by these cutter sets will be 290 minutes. In contrast, by considering all parts together, the total number of cutters needed is only 4, as

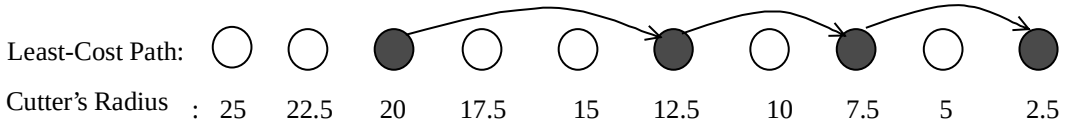


Figure 11: The Optimal Cutter Set Generated by considering all parts simultaneously (cutter loading time is 20 minutes)

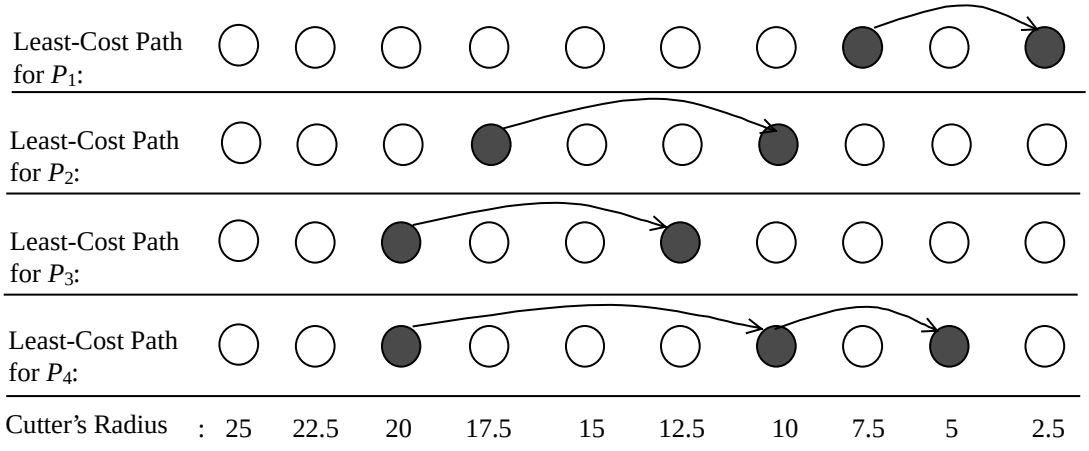


Figure 12: Optimal Cutter Sets for Individual Parts (cutter loading time is 20 minutes)

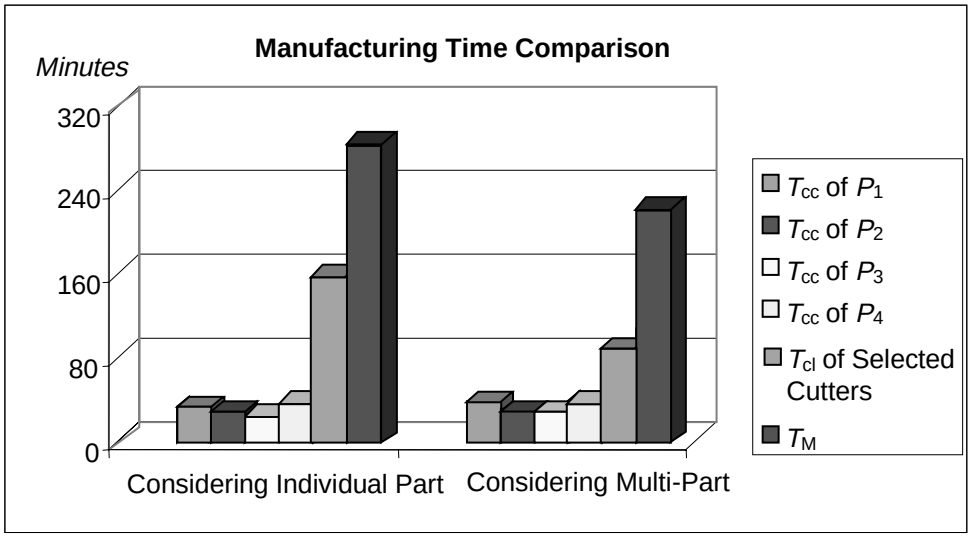


Figure 13: Comparison of Total Machining Time by Considering Four Parts Individually and Considering Four Parts Simultaneously

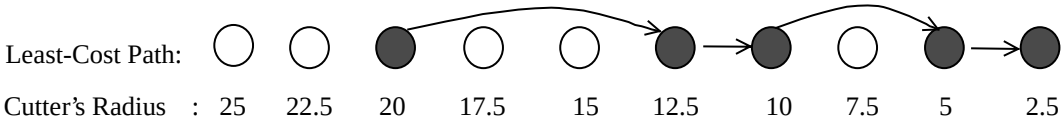


Figure 14: The Optimal Cutter Set Generated by considering all parts simultaneously (cutter loading time is 10 minutes)

shown in Figure 12 (their radii are 2.5 mm, 7.5 mm, 12.5 mm and 20 mm). As shown in Figure 13, the total machining time by using cutters selected by considering multi-part simultaneously will be 205 minutes. Thus, the total time saved by using multi-part cutter selection approach is (290-205)/290 = 29.3%.

Another interesting observation is that if the tool loading time changes, the optimal cutters may change. In particular, the lower the cutter loading time, the higher the total number of cutters in the optimal sequence may be. For example, as shown in Figure 14, if we take the previous example and change the cutter loading time to 10 minutes, then the number of cutters in optimal cutter set will be 5 rather than 4. Similarly, the higher the cutter loading time, the lower the total number of cutters in the optimal sequence. Meanwhile, the time saving will also be higher when considering multiple parts together compared to consider parts individually because the shared cutter loading time.

6. DISCUSSION AND CONCLUSION

In order to stay competitive in today's market, companies need to eliminate as many sources of manufacturing inefficiency as possible. One such source of inefficiency comes from unnecessary machine-tool reconfiguration operations.

In this paper, we describe a way to select an optimal set of cutting-tool sizes such that the cutting tools can be used for multiple different parts, thereby eliminating unnecessary machine tool reconfigurations. In particular, this paper describes the following new results:

1. We describe mathematical conditions for determining the region that can be covered by a given cutter, and discuss a problem with previous formulations of those conditions. Based on our conditions, we give an algorithm (not yet implemented) that can compute the coverable area exactly, and another algorithm (implemented) that can compute a close approximation.
2. We show how to represent the multi-part cutter selection problem as the problem of finding the least-cost path in a directed graph.
3. We describe a prototype implementation of our approach, and demonstrate it on an example. The example illustrates how significant savings can be achieved in the total machining time.

We plan to extend our work in the following areas to overcome current limitations:

1. We plan to implement the algorithm mentioned in Item 1 above.
2. Our current estimate of cutting time assumes that it is proportional to the ratio of the covered area and the cutter size. In practice, the cutting time will also depend on the cutter path. We plan to develop a better algorithm for estimating cutting time, by incorporating tool-path considerations.

3. Tool life plays an important role in tool selection. We plan to incorporate tool-life information in order to develop a more realistic estimate of total machining time.

4. So far, we have only considered geometric constraints in the cutter selection problem. We plan to extend our method to incorporate milling process constraints as well.

7. ACKNOWLEDGMENTS

This research has been supported in part by NSF grants DMI9896255 and DMI9713718, by AFRL grant F306029910013, and by a semester research award from the University of Maryland General Research Board. Opinions expressed in this paper are those of authors and do not necessarily reflect opinion of the funders.

8. REFERENCES

[1] S. Arya, S. W. Cheng and D. M. Mount. Approximation algorithm for multiple-tool milling. *Proc. Of the 14th Annual ACM Symposium on Computational Geometry*, pp. 297-306, 1998.

[2] M.Bala and T.C.Chang. Automatic cutter selection and optimal cutter-path generation for prismatic parts. *International Journal of Production Research*, 29(11), 2163-2176, 1991.

[3] T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms*, The MIT Press/McGraw Hill, 1990.

[4] C. M. Hoffmann. *Geometric and Solid Modeling: An Introduction*, Morgan Kaufman Publishers, 1989.

[5] Y. S. Lee, B. K. Choi and T. C. Chang. Cut distribution and cutter selection for sculptured surface cavity machining. *International Journal of Production Research*, 30(6), 1447-1470, 1993.

[6] K. Lee, T. J. Kim and S. E. Hong. Generation of toolpath with selection of proper tools for rough cutting process. *Computer-Aided Design*, vol(26) 822-831, Nov. 1994.

[7] Y. S. Lee and T. C. Chang. Application of computational geometry in optimization 2.5D and 3D NC surface machining. *Computers in Industry*, 26(1), 41-59, 1995.

[8] Y. S. Lee and T. C. Chang. Automatic cutter selection for 5-axis sculptured surface machining. *International Journal of Production Research*, 34(4), 977-998, 1996.

[9] T.Lim, J.Corney, J.M.Ritchie and D.E.R.Clark, Optimising automatic tool selection for 21/2D components. In *Proc. DETC 2000: 2000 ASME Design Engineering Technical Conference*, Baltimore, MD, September 10-13, 2000.

[10] T.Lim, J.Corney and D.E.R.Clark. Exact tool sizing for feature accessibility. *International Journal of Advanced Manufacturing Technology*, Vol.16, pp.791-802, 2000

[11] B. Mahadevan, L. Putta and S. Sarma. A feature free approach to tool selection and path planning in 3-axis rough cutting. *Proceedings of First International Conference on Responsive Manufacturing*, Nottingham, pp.47-60, September 1997.

[12] Ganping Sun, Fu-Chung Wang, Paul Wright and Carlo Sequin. Operation decomposition for freeform surface features in process planning. In *Proc. DETC 1999: 1999 ASME Design Engineering Technical Conference*, Las Vegas, Nevada, September 12-15, 1999.

[13] D. Veeramani, and, Y. S. Gau. Selection of an optimal set of cutting-tool sizes for 2.5D pocket machining. *Computer-Aided Design*, 29(12), 869-877, 1997.

[14] D.C.H. Yang and Z. Han. Interference detection and optimal tool selection in 3-axis NC machining of free-form surface. *Computer-Aided Design*, Vol.31, pp.303-315, 1999.

[15] Zhiyang Yao, S. K. Gupta and Dana Nau. A Geometric Algorithm for Finding the Largest Milling Cutter. *SME's Journal of Manufacturing Processes*. To appear, 2001.