ELSEVIER

Computer-Aided Design 35 (2003) 825–839

COMPUTER-AIDED DESIGN

www.elsevier.com/locate/cad

Algorithms for selecting cutters in multi-part milling problems

Zhiyang Yao[a], Satyandra K. Gupta[a,*], Dana S. Nau[b]

[a]*Department of Mechanical Engineering, Institute for Systems Research, University of Maryland, College Park, MD 20742, USA*
[b]*Department of Computer Science, Institute for Systems Research, University of Maryland, College Park, MD 20742, USA*

Received 11 December 2001; received in revised form 8 June 2002; accepted 10 June 2002

Abstract

This paper describes geometric algorithms for automatically selecting an optimal sequence of cutters for machining a set of 2.5-D parts. In milling operations, cutter size affects the machining time significantly. Meanwhile, if the batch size is small, it is also important to shorten the time spent on loading tools into the tool magazine and establishing z-length compensation values. Therefore, in small-batch manufacturing, 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. 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. In this paper we first present algorithms for finding the area that can be cut by a given cutter. Then we describe a graph search formulation for the tool selection problem. Finally, the optimal sequence of cutters is selected by using Dijkstra's shortest path planning algorithm.

Keywords: Geometric Algorithms; 2.5-D Milling; Cutter Selection

1. Introduction

Increasing emphasis on more personalized products and shrinking product lives is resulting in major changes in manufacturing practices [1]. 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 amount 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 the machine-tool were 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.

Human process planners and machine operators are already trying to create multi-use setups and machine-tool configurations and to exploit every opportunity for reusing tools and fixtures that have already been loaded into machine-tools. Here are two examples:

- In the sheet-metal industry, when given a new part, machine operators often analyze the previous machine-tool configurations to see how they can make use of portions of the existing configuration for the new part. In some cases, they even will intentionally plan configurations that will be useful for multiple parts.
- For CNC machining operations, operators often try to use the tools that are already loaded into the tool magazine. When they need to produce several different types of parts, they try to select a set of tools that can be used to produce all parts, and load all tools into the tool magazine before starting the machining operation for the first part.

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

* Corresponding author. Tel.: +1-301-405-5306; fax: +1-301-314-9477.
E-mail addresses: skgupta@eng.umd.edu (S.K. Gupta), yaodan@glue.umd.edu (Z. Yao), nau@cs.umd.edu (D.S. Nau).

0010-4485/03/$ - see front matter © 2003 Elsevier Science Ltd. All rights reserved.
PII: S0010-4485(02)00110-0

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 magazine) 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. 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 geometric algorithms 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.

Currently our algorithm is restricted to 2.5-D milling operations. In particular, we consider the problem of selecting a sequence of cylindrical cutters to cut all of the points in a 2.5-D target region without cutting any of the points in a 2.5-D obstruction region.

2. Related work

2.1. Multi-part process planning

Alva and Gupta [2] studied the problem of selecting shared bending punches. In sheet-metal bending, bends are formed using a combination of a punch and a die. These tools need to be able to withstand the bending forces, and their shapes should be such that there is no tool-part interference. The methodology for automatically synthesizing shapes of bending punches involves the following three steps:

1. Extract constraints on punch parameters, by performing intersection checks between geometric entities that define the parametric punch shape and geometric entities that define various intermediate workpiece shapes resulting during the bending process. Parametric geometric models of punches are used to describe the family of possible punch shapes. The resulting constraints on punch parameters are quadratic in nature for sash type (i.e. 2.5-D) parts.
2. Find a punch shape that does not intersect with any intermediate workpiece shape and has the maximum strength. For this, a combination of state-space search and mixed integer programming is used to try to find a punch shape that satisfies all intersection constraints generated in the previous step and maximizes the punch strength.
3. Verify that the designed punch can withstand stresses resulting from the bending forces.

In batch production environments for sheet-metal bending operations, press-brake configuration changes constitute a major portion of the production time. Gupta and Bourne have developed an algorithm for generating shared press-brake configurations [3]. The algorithm takes a family of parts and tries to find a shared configuration that can work for every part in the part family, using the following two steps:

1. Identify the tooling location and segment length constraints imposed by various bending operations in the part family. These constraints describe spatial constraints on lengths and locations of various tooling stages in the configuration. The resulting constraints are linear in nature.
2. Generate shared press-brake configurations that can satisfy all constraints generated in the previous step. For this, we use a combination of state-space search and incremental constraints propagation techniques. Any configuration that satisfies all tooling constraints is capable of accommodating every part in the part family.

2.2. Cutter selection

Several papers have been written that describe algorithms for solving cutter selection problem for 2.5-D milling process. To best of our knowledge, previous papers describe algorithms for solving single part problems. Moreover, most papers only consider cutting and tool change time. In practice, as product batch size shrinks, the tool loading time plays an increasingly important role in the total manufacturing time. Furthermore, with the development of high-speed tool changing mechanisms and twin spindle machine tools, the tool change time is increasingly playing less dominant role in the total machining time.

Bala and Chang presented a method to find the cutter set for prismatic parts [4]. In their approach, a finishing cutter is selected as equal to the smallest pocket corner radius. They select a rouging cutter as the largest cutter such that after machining with the roughing cutter, the remaining uncut area can be removed using a single pass of the finishing cutter.

Lee et al. extended Bala and Chang's work into cutter selection for 3D part [5–7]. In their work, a series of hunt planes are used. Each hunting plane intersects with the given 3D part and forms an intersection contour. Each contour is polygonized to acceptable polygons based on a given tolerance. For each hunting plane, a cutter that can be fitted into the 2-D geometric constraints among the vertices of the boundary polygons is selected. Following that, an optimal cutter set can be selected for adjacent hunting planes by merging the cutter selection to a single cutter to minimize tool changes and tool cost.

Lee et al. [8] assume that an efficient machining procedure is that use two tools in rough cutting, i.e. a bigger one is used for the portion with a simple shape while a small tool should be used for the complex portion. Octree method is used to find the near-optimal cutters.

Veeramani and Gau used a combination of Voronoi mountains and dynamic programming in cutter selection problem [9]. In their approach, the smallest cutter is the one that has the minimal radius of all the pocket corners. They use two stages to do the work: first using Voronoi mountain to get the relationship between the machinable area and the cutter size for closed pockets, then they use dynamic programming to find the optimal cutter set. Their dynamic programming formulation takes $O(n^3)$ running time in the worst case, where n is the number of tools considered. They use the Voronoi Mountain to estimate the area that can be cut by a cutter. This is done by generating the approximate cutter path length by using contour—parallel tool path. However, with the presence of circular edges, it is difficult to build the initial Voronoi Mountain. Moreover, if there are open edges, it is not clear how to build Voronoi mountains.

Mount et al. presented an approximation algorithm for finding the optimized multiple tools for milling process [10]. They transform the milling problem to a weighted set-cover problem using a greedy strategy to obtain a logarithmic ratio. Because of using Voronoi diagram in subdividing the milling domain, this approach has the same problem with Veeramani and Gau's approach when it encounters with open edges or circular edges.

Sun et al. considered cutter selection problem in their process planning problem [11]. One cutter is found according to the minimum of curvature radii, channel widths and corner radii. A second bigger cutter is found by trying several tools and selecting the one that minimizes the estimated machining time.

Yao et al. studied the problem of selecting one single maximal cutter that can cut the 'target region' without intersecting with the 'obstruction region' [12]. The definitions of target region and obstruction region are adopted in this paper because it can handle a general 2.5-D milling problem with both open and closed edges.

Recently a series of papers have been written that use traditional offset/inverse-offset approach to calculate the region that can be covered by a given cutter size [13–16]. As indicated in our previous paper [17], the result of using traditional offset/inverse-offset can introduce errors in the estimated area covered by the tool.

In our previous paper, we discovered the errors in using traditional offset/inverse-offset approach in finding the area a cutter can cover [17]. Based on our definitions and lemmas, we presented an algorithm for finding the approximate coverable area. We also presented an algorithm that can handle multi-part cutter selection problem in which each part only contains one single feature. Our current paper provides a new set of geometric algorithms that can (1) extract profiles from a part with multiple features, (2) exactly calculate the area a cutter can cover, (3) solve multi-part cutter selection problems in which each part may contain multiple features. We also provide detailed correctness proofs in this paper.

3. Problem formulation

3.1. Background and basic definitions

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 [12], 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 union of identically oriented 2.5-D solids and each 2.5-D solid is a machining feature that can be produced by one or more 2.5-D milling operations. In this case, the cutter selection problem can be reduced to a 2-D problem by considering cross-sections of these 2.5-D solids. For defining the cutter selection problem the following definitions are needed.

Definition 1. We define the region to be machined as the *target region* T (a region is a regular set of 2-D points). The target region need not be a connected set. For the part and stock shown in Fig. 1(a) and (b), and Fig. 1(c) shows an example of target region.

Definition 2. The *obstruction region* O is the region that the cutting tool should not cut during machining. The obstruction region need not be a connected set. For the part and stock shown in Fig. 1(a) and (b), and Fig. 1(c) shows an example of obstruction region.

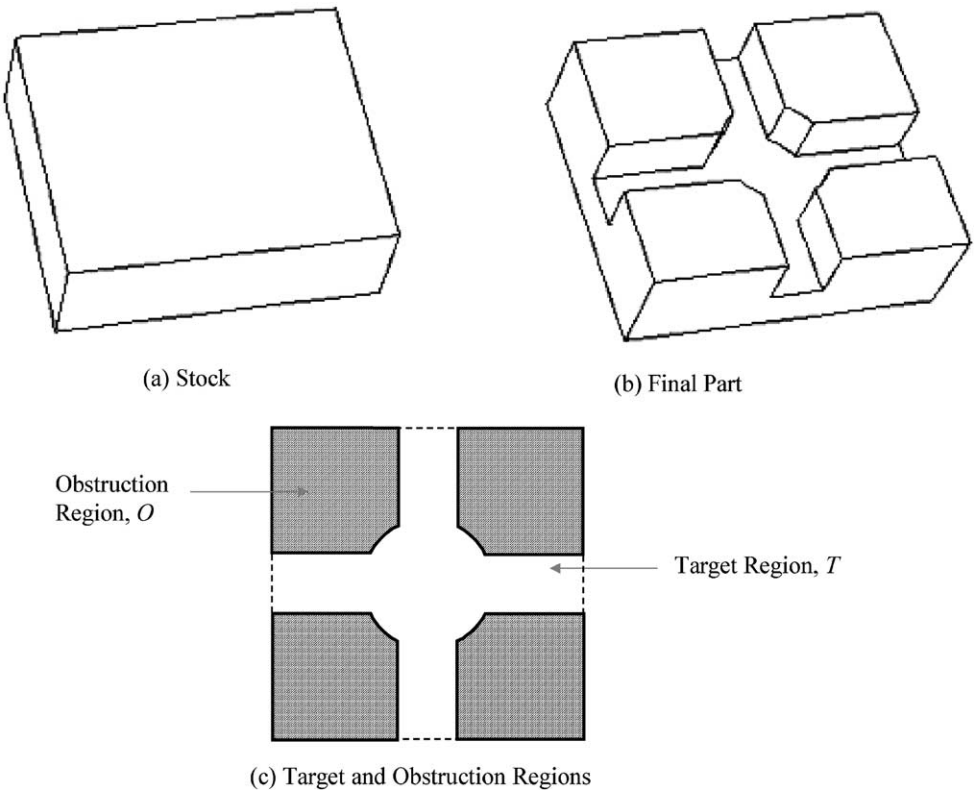


Fig. 1. Examples of the stock, final part, target region, and obstruction region.

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

Definition 3. 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 $R(C,p) = \{\text{all points } (u,v) \text{ such that } \sqrt{(u-x)^2 + (v-y)^2} \leq r(C)\}$. We will call $R(C,p)$ the set of points covered by C at p.

Definition 4. 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 $O \cap^* R(C,p) = \varnothing$.

Definition 5. 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.

Definition 6. The sub-region of a target region that can be safely covered by a given cutter is called *coverable region* and the area of coverable region is called *coverable area*.

In multi-cutter selection problems, multiple milling operations are used, each with a different milling cutter. After one part is loaded, the 2.5-D features of the part are machined one by one from top to bottom. For each 2.5-D feature, 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.

Definition 7. The total machining time T_M for the sequence of milling operations is the total time needed to machine features on the given set of parts. 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 spent 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 sec) compared to cutting time and cutter loading time (of the order of 5–10 min), in this paper we will ignore cutter change time. Therefore, in this paper we will use $T_M = T_{ct} + T_{cl}$.

To make this paper self-contained, we report the main Lemma's from our previous paper [17] (for proofs of these lemmas please see Ref. [17]):

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

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

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

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

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

3.2. Problem statement

We define the *multi-part cutter selection problem* as follows. Suppose 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 $(P_1, \ldots, P_L)$. In order to produce those parts, suppose we have a sequence of cutting tools $(C_1, C_2, \ldots, C_n)$, given in decreasing order of cutter radius (i.e. $r(C_1) > \cdots > r(C_n)$). Furthermore, suppose that C_n is small enough that it can safely produce all features of $P_1, \ldots, P_L$, and that for $m < n$, no C_m is small enough to safely produce all of $P_1, \ldots, P_L$. 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, this will minimize the total machining time T_M. In this paper we present an algorithm for solving this problem.

3.3. Overview of approach

There are basically there steps in our approach:

Step 1. Given a set of parts, we first extract the obstruction and target regions for each 2.5-D feature on each part. This step results in a set of features defined in terms of target regions and obstruction region. Section 4 describes this step in detail.

Step 2. For each combination of the cutter (from the set of available cutters) and the target region, we compute how much of the target region the 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. This problem is computationally similar to the problem of computing the offset for a 2-D point set, and previous approaches for this problem have been based on the use of the offsetting operators traditionally available in most solid modeling systems. However, in our previous paper we have shown that these approaches will not always produce correct results [17]. Therefore, we propose the open set offset definition and algorithm that can be used to calculate the coverable area precisely. Section 5 describes this step in detail.

Step 3. Once we have computed the coverable area for each combination of cutter and target region, 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. Section 6 describes this step in detail.

4. Algorithm for extracting target region and obstruction region

To select tools 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 Fig. 2, in which we have a 3D

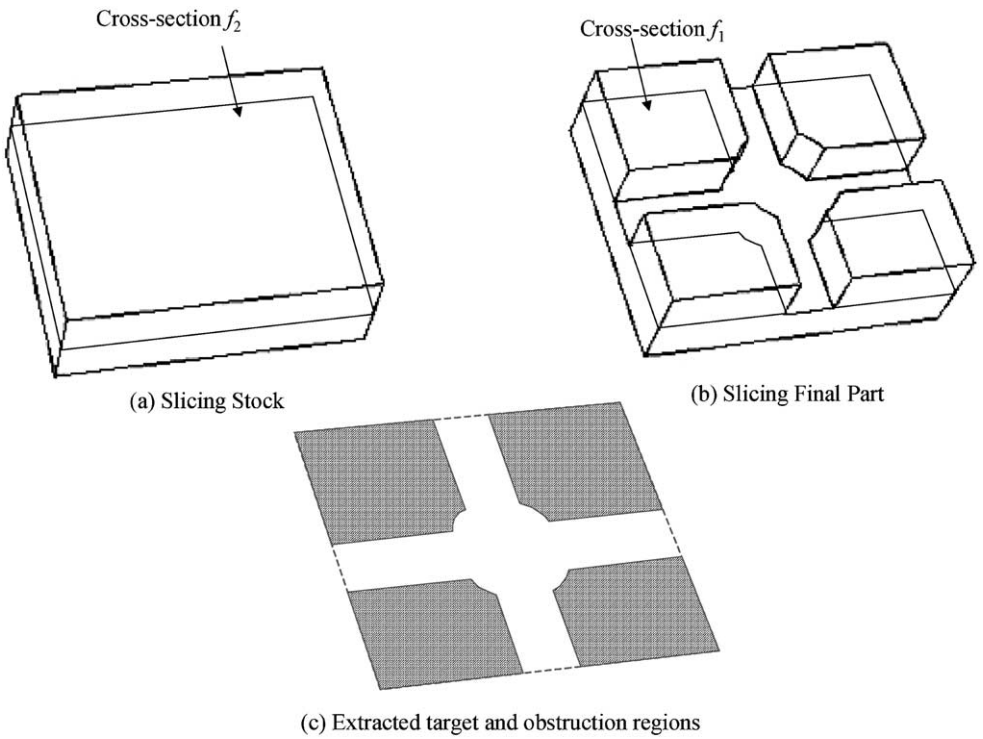


Fig. 2. Example of target and obstruction regions extraction (one feature).

model of a rectangular part whose faces are parallel to the *xy*, *yz*, and *xz* planes, and we also have its initial stock which is allocated in the same manner. Therefore, by subtracting the final part from its stock, we get its delta-volume, which is a single feature as shown in Fig. 2. This feature is a blind 2.5-D milling feature which is located in the part's top face. To find the target region and the obstruction region, we slice the stock and the part at the same *z*-value, which is at the bottom of the milling feature. Therefore, we can obtain two cross-sections that are parallel to the *xy* plane: a cross-section f_1 obtained by slicing the part, and a cross-section f_2 by slicing the stock. For this 2.5-D milling feature, the obstruction region is f_1, and the target region is $f_2 -^* f_1$.

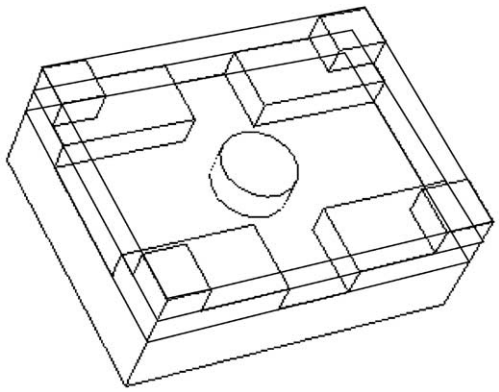
Most parts consist of multiple features. Therefore, it is important to automatically extract the profiles for all those features. As introduced in Section 3, in this paper, we assume that $S -^* P$ is a union of identically oriented 2.5-D solids and each 2.5-D solid is a machining feature that can be produced by one or more 2.5-D milling operations. Suppose we know the orientation of both the stock and the final part model, we can extract the target region and obstruction regions for each feature in the following algorithm.

Given a part P and its initial stock S, and feature orientation J the following algorithm is used to extract the target region and obstruction region for each feature:

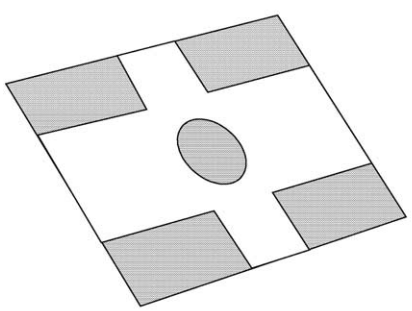
EXTRACT_TARGET_OBSTRUCTION_REGION (P, S, J)

1. Orient part and stock using J
2. Compute Delta Volume by $S -^* P$
3. Find set of planar faces F that are parallel to xy plane
4. Sort F by the z -value.
5. $i = 0$
6. While F has two or more faces
7. $\quad i = i + 1$
8. $\quad$ Find face f in F that has the smallest z-value
9. $\quad$ Form plane X_i using f
10. $\quad K_i = X_i \cap S$
11. $\quad T_i = K_i$ - interior(P)
12. $\quad O_i = K_i - T_i$
13. $\quad$ Sweep T_i upward to ensure its accessibility. If it is not accessible then this feature cannot be produced from this direction therefore stop
14. $\quad$ Remove f from F
15. End While
16. Return $((O_1,T_1), (O_2,T_2),\ldots, (O_i,T_i))$

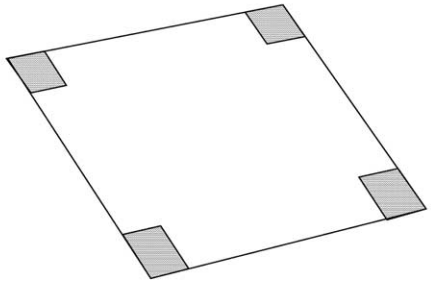
Fig. 3 shows a part whose initial stock shape is its rectangular bounding box. In this part, there are two features. Therefore, we first slice planes along three *z*-values as shown in Fig. 3(a). Then start from the lowest *z*-values, we first



(a) 3D part and slicing planes at different depth



(b) Obstruction and target region for the last cutting



(c) Obstruction and target region for the first cutting

Fig. 3. Example of profile extraction (two features).

extract the target and obstructions for the lowest feature as shown in Fig. 3(b), and then extract the target and obstructions for next feature as shown in Fig. 3(c).

5. Algorithms for finding coverable area for a given cutter

In order to solve the multi-part cutter selection problem, an important step is to find the coverable region and calculate the coverable area for each of the cutters $C_1, \ldots, C_n$. This section describes geometric algorithms for calculating the coverable area for a given feature and tool combination.

5.1. Main algorithm

Let B be the coverable region for given O, T and C. If we can get B and then we can easily compute the coverable area: $\alpha =$ area of B. Our algorithm for computing α is as follows:

COVERABLE_AREA_FINDING(C, O, T)

1. $(\bar{A}_1(O,C), \bar{A}_2(O,C))$= Call COMPUTE_ $\bar{A}(C, O, T)$ (see the discussion of this below)
2. E_1 = Call Offset $(\bar{A}_1(O,C)$, $r(C))$
3. E_2 = Call Offset $(\bar{A}_2(O,C), r(C))$
4. $E = E_1 \cup^* E_2$
5. $Q = T -^* E$
6. $B = T -^* Q$
7. Return α = the area of B

Correctness of COVERABLE_AREA_FINDING Algorithm. Theorem 1 establishes that the COVERABLE_AREA _FINDING algorithm correctly computes the coverable area.

Theorem 1. *B is the set of coverable points for C.*

Proof. As proved in Theorem 2, the union of $\bar{A}_1(O, C)$ and $\bar{A}_2(O, C)$ produces $\bar{A}$, therefore, $E_1 \cup {}^*E_2$ produces E.
In Step 5, $Q = T -^* E$. Since Q is a subset of $\bar{E}$, from Lemma 4, we conclude that for every $p \in Q$, there is no permissible location for C to cover p.
In Step 6,$B = T -^* Q$. B is a subset of E. Therefore, from Lemma 3, we conclude that for every $p \in B$, there is a permissible location for C to cover p.
Therefore, B is the set of coverable points for C. □

Fig. 4 shows an example in which A, $\bar{A}$, E, $\bar{E}$, B, and Q are given.

5.2. Algorithm for calculating $\bar{A}$

In this section, we give an algorithm to compute Step 1 of COVERABLE_AREA_FINDING(C, O, T). In the algorithm, we save $\bar{A}$ by two items. The first item is a regularized 2-D set $\bar{A}$ and the second item is a lower dimensional 1D and 0D elements set $\bar{A}_2$. Steps 2 and 3 in COVERABLE_AREA _FINDING(C, O, T) is to find the traditional offset of $\bar{A}_1$ and $\bar{A}_2$, which can be easily obtained by using traditional 'offset' operator and will result in two regular sets. Then we can use regular operator to union those offset sets together to get the correct result.

In order to do so, we also need the following definitions.

Definition 8. Given a closed set S, we define its *open offset* by distance r as $\mathscr{E}(S, r) = \{p | \exists q \in S,$ such that $d(q, p) < r\}$.

Definition 9. Given a closed set S, we define its *exact offset boundary* by distance r as $\mathscr{L}(S, r) = \{p | d(p, S) = r\}$, where $d(p, S) = \min\{d(p, q),$ for $\forall q \in S\}$.

Fig. 5 shows the example of open offset and the exact offset boundary when the set S contains a single point.

From the above, it follows that $\mathscr{L}(S, r) = \mathscr{F}(S, r) - \mathscr{E}(S, r)$, where $\mathscr{F}(S, r)$ is the traditional offset operator which is defined by $\mathscr{F}(S, r) = \{p | \exists q \in S,$ such that $d(q, p) \leq r\}$.

Given a cutter C, target region T, and obstruction region O, our algorithm to compute $\bar{A}_1$ and $\bar{A}_2$ works in the following manner:

COMPUTE_Ā (C, O, T)

1. Initial set $\mathscr{A}(O, r)$= NULL, $\bar{A}_1$ =NULL, $\bar{A}_2$ = NULL, U = the bounding box of the part
2. Compute the traditional offset of the obstruction region O as $\mathscr{A}(O, r)$
3. For each line or arc segment l_i of the boundary of O
4. Compute and save the closed offset of l_i as $\mathscr{A}(l_i, r)$ and the exact offset boundary of l_i as $\mathscr{L}(l_i, r)$ //In the following section, we introduce methods to compute $\mathscr{L}(l_i, r)$
5. End For
6. For each l_i
7. Compute $\mathscr{H}(l_i,r)$ = portion of $\mathscr{L}(l_i, r)$ that is not inside of $\mathscr{A}(O, r)$ and $\mathscr{A}(l_j, r)$, for any $j \neq i$
8. Add $\mathscr{H}(l_i,r)$ in $\bar{A}_2$
9. End For
10. $\bar{A}_1 = U -^* \mathscr{A}(O, r)$
11. Return $(\bar{A}_1, \bar{A}_2)$

All of the above steps are straight forward except for Step 4, which computes the exact offset boundary of l_i. For an arbitrary geometry, it is hard to find its exact offset boundary. However, in this paper, only line or arc

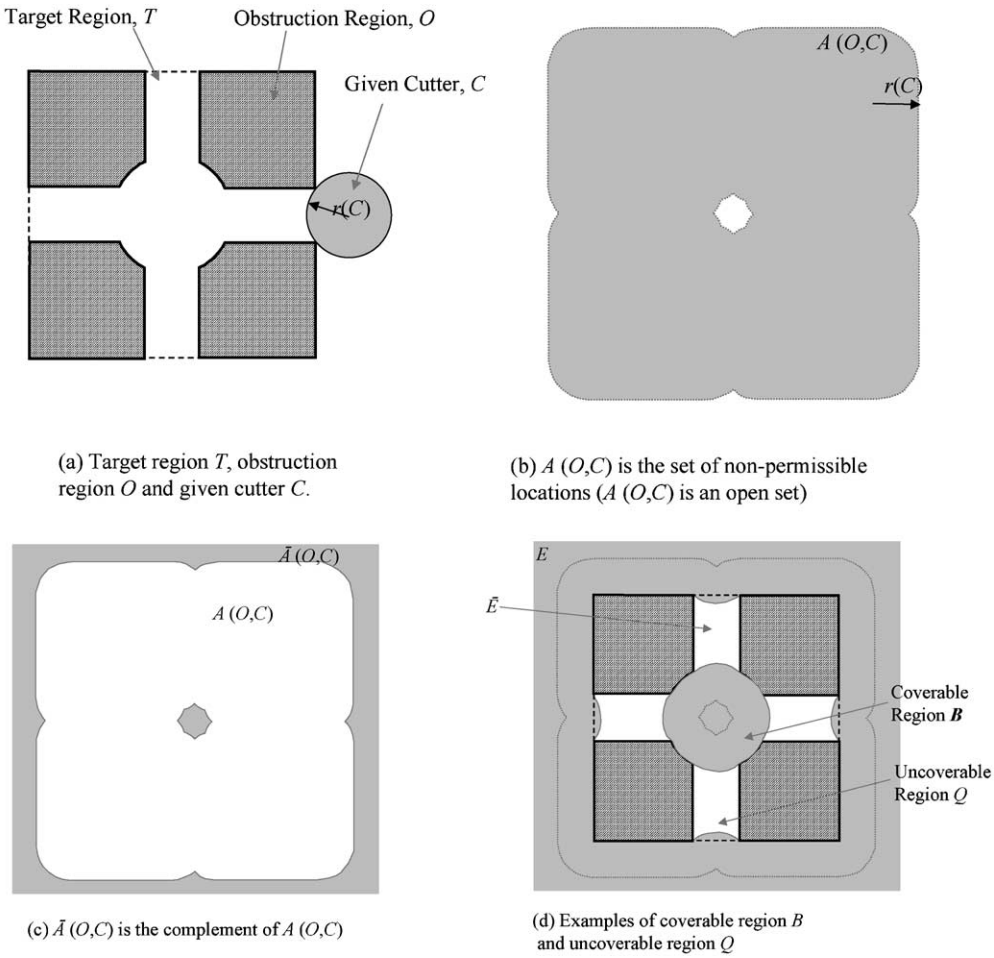


Fig. 4. Illustration of definitions.

segments are considered. Therefore, we can derive equations to calculate the exact offset boundaries for those special cases.

As shown in Fig. 6(a), the exact offset boundary for a line segment l is given by: $\mathscr{L}(l,r) = \mathscr{F}(l,r) - \mathscr{E}(l,r) = b(\mathscr{F}(l,r))$. Here b refers to the boundary.

For an arc segment a, there are three cases:

1. If the offset distance is less than or greater than the radius of a, then its exact offset boundary is: $\mathscr{L}(a,r) = \mathscr{F}(a,r) - \mathscr{E}(a,r) = b(\mathscr{F}(a,r))$. Fig. 6(b) and (c) show examples of open offset in these cases.

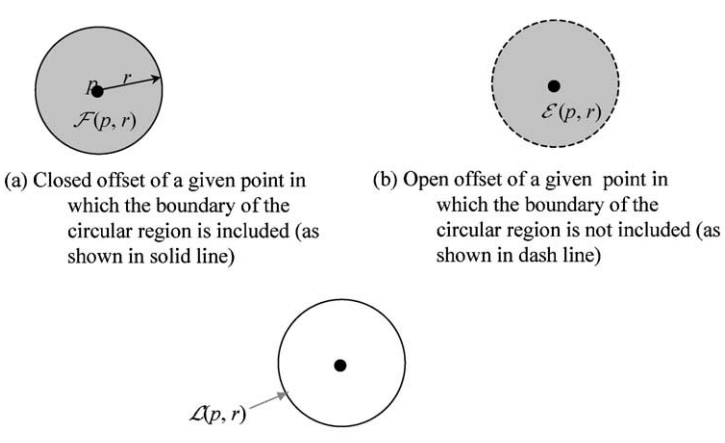


Fig. 5. Examples of open offset, closed offset and exact offset boundary of a given point.

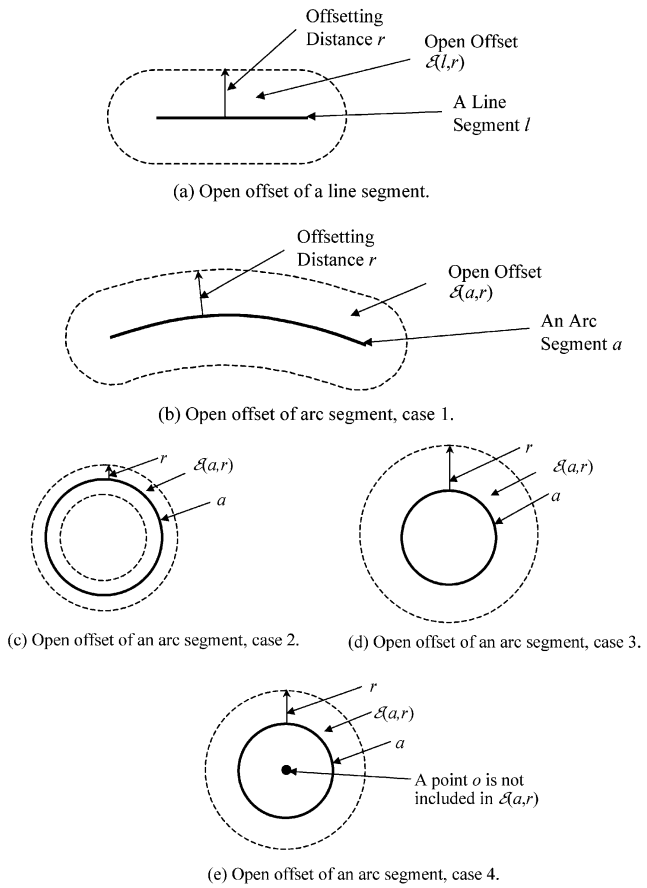


Fig. 6. Different cases of open offset for line and arc segments.

2. If the offset distance is equal to the radius of a and a is not a full circle, then its exact offset boundary is $\mathscr{L}(a,r) = \mathscr{F}(a,r) - \mathscr{E}(a,r) = b(\mathscr{F}(a,r))$. Fig. 6(d) shows an example of this case.
3. If the offset distance is equal to the radius of a and a is a full circle, and suppose its center point is o, then its exact offset boundary is $\mathscr{L}(a,r) = \mathscr{F}(a,r) - \mathscr{E}(a,r) = b(\mathscr{F}(a,r)) \cup o$. Fig. 6(e) shows an example for this case.

Correctness of Algorithm COMPUTE_$\bar{A}$. In the following paragraphs, we will show that the union of $\bar{A}_1$ and $\bar{A}_2$ by our algorithm corresponds to mathematical definition of $\bar{A}$.

Lemma 5. *The shortest distance from any point in $\bar{A}_2$ to O is exactly r.*

Proof. For any point p, $p \in \bar{A}_2$, suppose q is the point in O such that $d(p,q)$ is the shortest distance from p to O, thus q is on the boundary of O. If $d(p,q) > r$, then there is no l_i such that $p \in \mathscr{L}(l_i, r)$, that leads to contradiction. If $d(p,q) < r$, then there must be a l_i such that $p \in \mathscr{E}(l_i, r)$, then $p \notin \bar{A}_2$, that also lead to contradiction. Therefore, $d(p,q) = r$. □

Lemma 6. *Any point that is in $\bar{A}$ will be in $\bar{A}_1$ or in $\bar{A}_2$.*

Proof. As definition, we know that $\bar{A} = \{p : \forall q \in O, d(p,q) \geq r(C)\}$. For any point p, $p \in \bar{A}$, we can find a point q, $q \in O$, and $d(p,q)$ is minimal for all points in O. It is straight forward that q must be on the boundary of O. If $d(p,q) > r(C)$, as $\mathscr{F}(O,r)$ is defined as $\mathscr{F}(O,r) = \{p | \exists q \in O$, such that $d(q,p) \leq r\}$, therefore, $p \in \bar{A}_1$. If $d(p,q) = r(C)$, as shown in Lemma 5, $p \in \bar{A}_2$. □

Lemma 7. *Every point in $\bar{A}_1$ will belong to $\bar{A}$.*

Proof. For any point p, $p \in \bar{A}_1$, we know that for any point q, $q \in O$, $d(p,q) \geq r$, therefore, $p \in \bar{A}$. □

Lemma 8. *Every point in $\bar{A}_2$ will belong to $\bar{A}$.*

Proof. From Lemma 5, it is straight forward that Lemma 8 is correct. □

Theorem 2. *The union of $\bar{A}_1$ and $\bar{A}_2$ produces $\bar{A}$.*

Proof. Lemmas 6–8 together prove that $\bar{A}$ is the union of $\bar{A}_1$ and $\bar{A}_2$. □

Theorem 2 proves that algorithm COMPUTE_$\bar{A}$ is correct.

Fig. 7 shows the example of calculating $\bar{A}_1$ and $\bar{A}_2$.

6. Algorithm for 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.

Recall from Section 3 that we are given a sequence of cutting tools $(C_1, C_2, \ldots, C_n)$, listed in decreasing order of cutter radii; 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.

Definition 10. We define the *workpiece state* Γ_{ik} as follows. For $k = 1, \ldots, L$, let $\Gamma_{0k} = S_k$, and for $i = 1, \ldots, n$, let Γ_{ik} 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_k as it can safely cut.

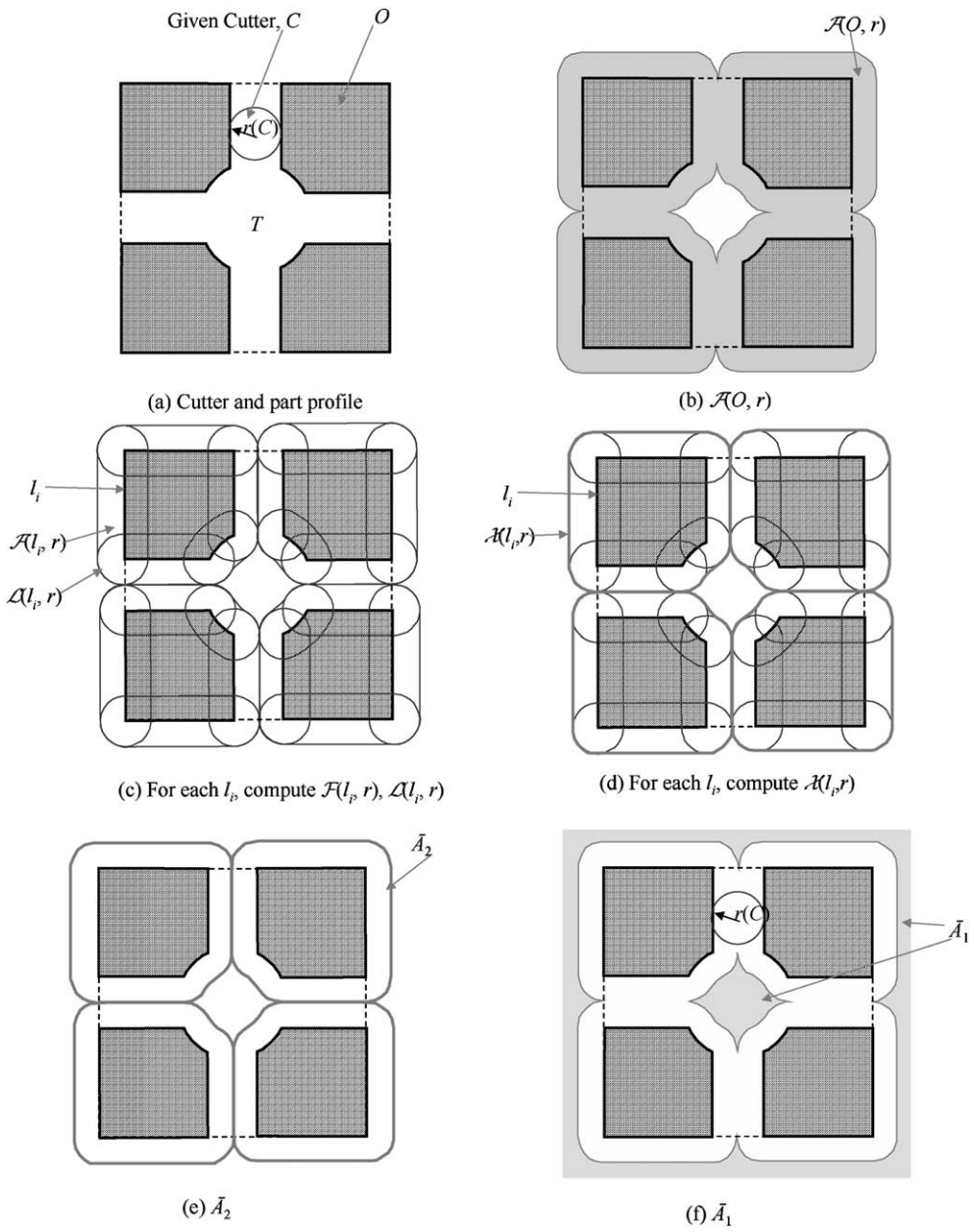


Fig. 7. Example of computing $\bar{A}_1$ and $\bar{A}_2$.

From this definition, it follows that for every $i > 0$, Γ_{ik} is equal to the set of points in T_k 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_k that they can safely cut is a subset of the portion of T_k that C_i can safely cut.

Definition 11. 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 $B_{ik} = T_k -^* G_{ik}$, and let A_{ik} be the area of B_{ik}. (As a special case, note that $B_{0k} = T_k -^* G_{0k} = T_k -^* S_k = \varnothing$, and thus $\alpha_{0k} = 0$.) Then the safely coverable area for the composite state Γ_i using cutter C_i is given by

$$\alpha_i = \sum_{k=1}^{L} \alpha_{ik}$$

With those definitions in mind, we give the following algorithm in finding the optimal sequence of cutters.

MULTIPART_CUTTER_SELECTION($(C_1,C_2,\ldots,C_n)$, $(P_1, \ldots, P_L)$, $(S_1, \ldots, S_L)$, T_{cl}, h)

1. Initialize G as empty
2. For i =1 to n
3. For j = 1 to L,
4. Call EXTRACT_TARGET_OBSTRUCTION_REGION (P_j, S_j), store $((O_{j1}, T_{j1}), (O_{j2}, T_{j2}), \ldots,(O_{j\,Nj}, T_{j\,Nj}))$, N_j is the number of 2½D features on part P_j
5. For k = 1 to N_j
6. α_{ijk} = COVERABLE_AREA_FINDING (C_i, O_j, T_j)
7. $\alpha_{ij} = \sum_{k=1}^{Nj} \alpha_{ijk}$
8. End For
9. $\alpha_i = \sum_{j=1}^{L} \alpha_{ij}$
10. End For
11. Insert node Γ_i in graph G with corresponding value α_i and cutter C_i
12. End For
13. For i =1 to n
14. For j = 1 to n
15. $w(\Gamma_i,\Gamma_j) = T_{cl} + h \times (\alpha_j - \alpha_i)/r_j$
16. End For
17. End For
18. Call Dijkstra's algorithm on G, return the least-cost path in G from Γ_0 to Γ_n.

Lemma 9. *Cutter sequence associated with any path in G is a valid sequence of cutters.*

Proof. Since Γ_0 represents the initial stocks $(S_1, \ldots, S_L)$ and Γ_n represents the final parts $(P_1, \ldots, P_L)$. In the graph, any valid path starting from Γ_0 to Γ_n represents a cutting sequence in which the final parts can be produced form the initial stocks. Therefore, any cutter sequence associated with a path in G is a valid sequence. □

Lemma 10. *Minimum T_M of using cutter sequence associated with a path in graph G is equal to the cost of the path in G.*

Proof. G is a 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. Each edge (Γ_i, Γ_j) in G is assigned a cost $w(\Gamma_i, \Gamma_j) = T_{cl} + h \times (\alpha_j - \alpha_i)/r_j$, where T_{cl} is the cutter loading time (usually determined experimentally) and $h \times (\alpha_j - \alpha_i)/r_j = T_{ct}$, where h is a factor determined by machining parameters.[1] Since the size of cutter C_j is smaller than the size of cutter C_i, if any portion of the coverable area of C_i is left to be cut by cutter C_j, the cutting time will increase according to the cutting time estimation equation used in this paper. Therefore, $h \times (\alpha_j - \alpha_i)/r_j$ represents the least cutting time of using cutter C_j right after cutter C_i is used. Thus, the cost of any path in G corresponds to the minimum T_M of using cutter sequence associated with the path. □

Theorem 3. *The least-cost path in G gives the sequence of cutters that minimizes T_M.*

Proof. Lemmas 9 and 10 together prove that the least-cost path in G gives the sequence of cutters that minimizes T_M. □

[1] Since we know the region we can also estimate cutter path length by actually computing the trajectory based on zigzag or contouring strategies. Many commercial systems such as Pro/Engineer® can generate cutting tool path for general pockets.

Table 1
Coverable area/cutter size table of example 1

Cutter size	Cuttable area				
	P1	P2	P3	P4	All parts
5	15012.4	12145.1	10510.0	22040.0	59707.5
7.5	15012.4	12145.1	9139.3	19285.7	55582.5
10	13020.0	1092.6	7210.8	16204.4	37527.8
12.5	11920.5	1092.6	6753.1	15087.0	34853.2
15	6120.0	9510.4	6102.5	10936.0	32668.9
17.5	5115.2	9146.6	5749.0	9704.5	29715.3
20	455.1	8760.7	5392.2	9252.0	23860.0
22.5	390.6	8475.5	0	6984.6	15850.7
25	285.8	8324.0	0	4254.4	12864.2
27.5	214.0	7862.0	0	2895.0	10971.0
30	103.0	0	0	1375.3	1478.3
32.5	81.5	0	0	842.2	923.7

Using Dijkstra's algorithm, this least-cost path can be found in time $O(n^2)$ [18]. n is the total number of given cutters.

7. Implementation and examples

We have implemented our algorithm, using C++ and the ACIS® kernel. Following are two examples.

7.1. Example 1: parts each has one single feature

Fig. 11 shows parts P_1, P_2, P_3 and P_4, each part having only one single feature. In this example, we are given 12 cutters $(C_1, \ldots, C_{12})$ and their radii are 5, 7.5, 10, 12.5, 15, 17.5, 20, 22.5, 25, 27.5, 30 and 32.5 mm.

Table 1 shows the cutter size and cuttable area relationship for parts shown in Fig. 8. Table 2 shows the selected optimal sequence of cutters and corresponding total machining time under different cutting conditions. If the cutter loading time is 10 min, the optimal sequence of cutters by consider multi-parts together will be 5, 10, 17.5, 22.5 and 27.5 mm. The total machining time is 82 min. Compared to the total machining 89 min by consider cutter selection for individual part, the total time saving is 8%. If the cutter loading time changes to 20 min, then the selected optimal sequence of cutters is 5, 12.5, 20 and 27.5 mm. The total machining time will be 120, and the total saving will be 9%.

7.2. Example 2: parts each has two features

Figs. 9–11 show parts P_5, P_6, and P_7, each part having two features. In this example, we are given 10 cutters $(C_1, \ldots, C_{10})$ and their radii are 1, 2.5, 5, 10, 15, 20, 25, 30, 40 and 50 mm.

Table 3 shows the cutter size and cuttable area relationship for these three parts. Table 4 shows the selected optimal sequence cutters and corresponding total machining time under different cutting conditions. If the cutter loading time is 10 min, the optimal sequence of cutters by considering multi-parts together will be 2.5, 5, 25, and 40 mm. The total machining time is 74 min. Compared to the total machining 81 min by consider cutter selection for individual part, the total time saving is 8%. If the cutter loading time changes to 20 min, then the selected optimal sequence of cutters is 2.5, 10, and 30 mm. The total machining time will be 101, and the total saving will be 10%.

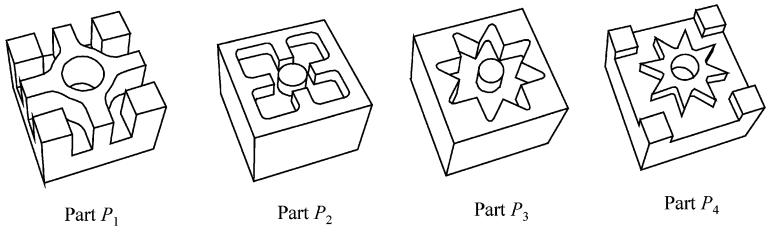


Fig. 8. Example 1: Parts each has only one feature.

Table 2
Experimental results of example 1

Cutting condition	Optimal sequence of cutters (mm)	Total machining time (consider multi-parts)	Anticipated total machining time saving compare to selecting cutters individually (%)
Cutter loading time is 10 min	5, 10, 17.5, 22.5, 27.5	82	8
Cutter loading time is 20 min	5, 12.5, 2, 27.5	120	9

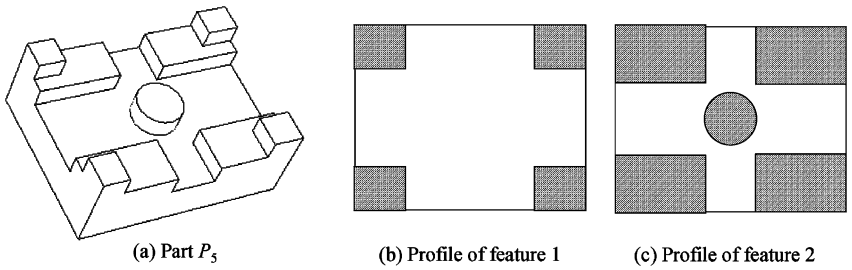


Fig. 9. Example Part P_5.

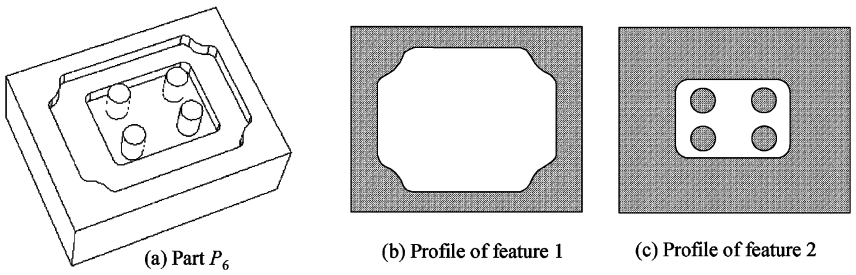


Fig. 10. Example Part P_6.

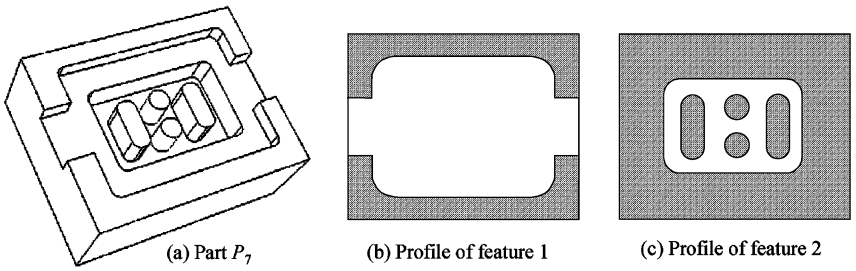


Fig. 11. Example Part P_7.

7.3. Observations

From these two examples, we can obtain the following observations:

1. 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. The experimental results indicate that this statement is correct.
2. Compared to selecting cutters for part individually, this multi-part cutter selection approach produces better cutter sequence, and thus leads to shorter total machining time because of the shared cutter loading time.
3. If the tool loading time changes, the optimal cutters may change. In particular, the lower the cutter loading time,

Table 3
Coverable area/cutter size table of example 2

Cutter size	Cuttable area			
	Part 5	Part 6	Part 7	All parts
1	18403.2	15708.0	15964.2	50075.4
2.5	18403.2	15708.0	15964.2	50075.4
5	18403.2	15708.0	15157.8	47348.0
10	16637.4	13334.8	13379.4	43351.6
15	15364.8	12920.0	12897.0	41181.8
20	14812.2	12141.4	12285.0	39238.6
25	13957.2	11407.0	11167.2	36531.4
30	10836.0	10650.5	10589.4	32075.9
40	997.2	8415.0	10101.6	28423.8

Table 4
Experimental results of example 2

Cutting condition	Optimal sequence of cutters (mm)	Total machining time (consider multi-parts)	Anticipated total machining time saving compare to selecting cutters individually (%)
Cutter loading time is 10 min	2.5, 5, 25, 40	74	8
Cutter loading time is 20 min	2.5, 10, 30	101	10

the higher the total number of cutters in the optimal sequence may be. Similarly, the higher the cutter loading time, the lower the total number of cutters in the optimal sequence.

8. Discussion and conclusions

8.1. Summary

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 give the definition and an algorithm for computing open offset. This helps in computing coverable area exactly as opposed to the approximation approach proposed in our previous paper [17]. This is accomplished by using very simple operators. Based on our conditions, we give an algorithm that can compute the coverable area exactly.
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.

8.2. Anticipated impact

The proposed approach presents a significant improvement over the current single-part tool-selection approaches. Machine-tool reconfigurations constitute a major portion of the production time in the batch production environment. Multi-part tool selection can be used to significantly cut down the total number of reconfigurations and increase the overall throughput. In particular, we believe that the proposed research will serve as the enabling technology for the following:

- The ability to handle the larger part varieties that result from mass customization. The advent of mass customization is resulting in a larger variety of parts on the shop floors. By sharing tools and machine-tool configurations, it will be possible to handle a larger variety of parts on the shop-floor without significantly lowering the overall throughput.
- More flexible scheduling to reduce in-process inventory. Currently, machine-tools are set up to perform operations on single parts. This requires that all parts of a particular type be completed before moving on to the next type of part. Unfortunately, the assembly process cannot start until all different types of parts that are needed for assembly are completed. This leads to large in-process inventory. The ability to have shared configurations, then would give factories much more flexibility in scheduling, and thereby reduce the in-process inventory.

8.3. Future work

Our work can be extended in the following areas to overcome current limitations:

1. 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.
2. 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.
3. 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.

Acknowledgments

This research has been supported by NSF grant DMI9896255, DMI0093142, NSF Equipment Grant EIA9986012, and ONR grant N000140010416. Opinions expressed in this paper are those of authors and do not necessarily reflect opinion of the sponsors.

References

[1] Pine BJ. Mass customization: the new frontier in business competition. Harvard: Harvard Business School Press; 1993.
[2] Alva U, Gupta SK. Automated design of sheet metal punches for bending multiple parts in a single setup. Robotics Comput Integrated Manufact 2001;17(1/2):33–47.
[3] Gupta SK, Bourne DA. Sheet metal bending: generating shared setups. ASME J Manufact Sci Engng 1999;121:689–94.
[4] Bala M, Chang TC. Automatic cutter selection and optimal cutter-path generation for prismatic parts. Int J Production Res 1991;29(11): 2163–76.
[5] Lee YS, Choi BK, Chang TC. Cut distribution and cutter selection for sculptured surface cavity machining. Int J Production Res 1993; 30(6):1447–70.

[6] Lee YS, Chang TC. Application of computational geometry in optimization 2.5D and 3D NC surface machining. Comput Ind 1995; 26(1):41–59.
[7] Lee YS, Chang TC. Automatic cutter selection for 5-axis sculptured surface machining. Int J Production Res 1996;34(4):977–98.
[8] Lee K, Kim TJ, Hong SE. Generation of toolpath with selection of proper tools for rough cutting process. Comput-Aided Des 1994; 26(11):822–31.
[9] Veeramani D, Gau YS. Selection of an optimal set of cutting-tool sizes for 2.5D pocket machining. Comput-Aided Des 1997;29(12): 869–77.
[10] Arya S, Cheng SW, Mount DM. Approximation algorithm for multiple-tool milling. Proceedings of the 14th Annual ACM Symposium on Computational Geometry, 1998. p. 297–306.
[11] Sun G, Wang F-C, Wright P, Sequin C. Operation decomposition for freeform surface features in process planning. Proceedings of the DETC 1999: 1999 ASME Design Engineering Technical Conference. Las Vegas, Nevada, 1999. p. 12–5.
[12] Yao Z, Gupta SK, Dana N. A geometric algorithm for finding the largest milling cutter. J Manufact Process 2001;3(1):1–16.
[13] Mahadevan B, Putta L, Sarma S. A feature free approach to tool selection and path planning in 3-axis rough cutting. Proceedings of First International Conference on Responsive Manufacturing, Nottingham, 1997. p. 47–60.
[14] Lim T, Corney J, Ritchie JM, Clark DER. Optimising automatic tool selection for 21/2D components. Proceedings of DETC: 2000 ASME Design Engineering Technical Conference, Baltimore; September 10–13 2000.
[15] Lim T, Corney J, Clark DER. Exact tool sizing for feature accessibility. Int J Adv Manufact Technol 2000;16:791–802.
[16] D'Souza R, Wright P, Séquin C. Automated microplanning for 2.5-D pocket machining. J Manufact Syst 2001;20(4):288–96.
[17] Yao Z, Gupta SK, Nau D. A geometric algorithm for selecting optimal set of cutters for multi-part milling. ACM Solid Modeling 2001.
[18] Cormen TH, Leiserson CE, Rivest RL. Introduction to algorithms. Cambridge/New York: The MIT Press/McGraw Hill; 1990.

Zhiyang Yao is a PhD student in Mechanical Engineering Department in University of Maryland, College Park. He received a BS in 1995 and an MS in 1998 in Mechanical Engineering from Tsinghua University, People's Republic of China. His research interests are Computer-Aided Design and Manufacturing, Concurrent Engineering, Geometric Reasoning. Particularly, he is working on constructing innovative process plan that can significantly reduce time-to-market and enable cost effective small batch manufacturing. He has authored or co-authored 14 articles in journals, conference proceedings, and technical reports. He is a student member of ASME.

Satyandra K. Gupta is an Associate Professor in Mechanical Engineering Department and the Institute for Systems Research. He received a PhD in Mechanical Engineering from the University of Maryland at College Park in 1994. Prior to joining the University of Maryland, he was a Research Scientist in the Robotics Institute and an Adjunct Assistant Professor of Manufacturing in the Graduate School of Industrial Administration at Carnegie Mellon University. The objective of Dr Gupta's research is to develop new algorithms for creating next-generation computer-aided design and manufacturing (CAD/CAM) systems that can reduce time-to-market, enable cost effective small batch manufacturing, and facilitate manufacturing of geometrically complex heterogeneous objects. Dr. Gupta is a member of American Society of Mechanical Engineers (ASME) and Society of Manufacturing Engineers (SME). He has authored or co-authored more than eighty articles in journals, conference proceedings, and book chapters. He has organized several conference sessions in the area of computer-aided design and manufacturing. He has served as Program Co-Chair in 1998 ASME's Design for Manufacturing Conference, Papers Chair in 1999 ASME's Design for Manufacturing Conference, and Exhibit Chair in 2000 ASME's Design Engineering Technical Conferences. Dr Gupta has won many honors and awards for his research contribution to computer-aided design and manufacturing area. In particular, he has been awarded a Graduate School Fellowship and an Institute for Systems Research Graduate Fellowship during his PhD study at University of Maryland. Other awards received by Dr Gupta include Best Paper Award in ASME's International Conference on Computers in Engineering in 1994, Best Paper Award in ASME's Design for Manufacturing Conference in 1999, ONR's Young Investigator Award in 2000, SME's Robert W. Galvin Outstanding Young Manufacturing Engineer Award in, NSF's CAREER Award in 2001.

Dana S. Nau is a professor at the University of Maryland, with a joint appointment in the Department of Computer Science and the Institute for Systems Research, and affiliate appointments in the Institute for Advanced Computer Studies and the Department of Mechanical Engineering. His research interests include AI planning and searching, and computer-integrated design and manufacturing. He received his PhD from Duke University in 1979, where he was an NSF graduate fellow. He has more than 250 technical publications. He has received an NSF Presidential Young Investigator award, the ISR Outstanding Faculty award, and several 'best paper' awards. He is a Fellow of the American Association for Artificial Intelligence (AAAI).