Approximation Algorithms for Multiple-Tool Milling

Sunil Arya* Siu-Wing Cheng†
Department of Computer Science
The Hong Kong University of Science & Technology
Clear Water Bay
Hong Kong

David M. Mount‡
Department of Computer Science &
Institute for Advanced Computer Studies
University of Maryland
College Park, Maryland

Abstract

Milling is the mechanical process of removing material from a piece of stock through the use of a rapidly spinning circular milling tool in order to form some desired geometric shape. An important problem in computer-aided design and manufacturing is the automated generation of efficient milling plans for computerized numerically controlled (CNC) milling machines. Among the most common milling problems is simple 2-dimensional pocket milling: cut a given 2-dimensional region down to some constant depth using a given set of milling tools. Most of the research in this area has focused on generating such milling plans assuming that the machine has a tool of a single size. Since modern CNC milling machines typically have access to a number of milling tools of various sizes and the ability to change tools automatically, this raises the important optimization problem of generating efficient milling plans that take advantage of this capability to reduce the total milling time.

We consider the following *multiple-tool milling problem*: Given a region in the plane and a set of tools of different sizes, determine how to mill the desired region with minimum cost. The problem is known to be NP-hard even when restricted to the case of a single tool. In this paper, we present an efficient approximation algorithm for the multiple-tool milling problem. The running time and approximation ratio of our algorithm depend on the simple cover complexity of the milling region.

*Research supported in part by RGC CERG HKUST 736/96E.
†Research supported in part by RGC CERG HKUST160/95E.
‡Research supported in part by the National Science Foundation under grant CCR–9712379.

1 Introduction.

Milling is one of the most important methods used in the manufacturing of mechanical parts in computer-aided manufacturing (CAM). It is applied to a *workpiece* of (typically metal) stock sometimes called the *part* or *billet*, which is clamped to a moving platform that is then translated under a rapidly spinning circular-shaped *milling tool*. It is somewhat more natural to think of the stock as remaining stationary and the tool translating above it. In this way material is removed, or *milled*, from the part. The overall problem is how to construct milling plans in order to achieve a given final geometric shape within the shortest amount of time.

There are several kinds of milling depending on the numbers of degrees of freedom possessed by the tool relative to the workpiece. In this paper we focus on the simplest case, but one that is common in practice, where continuous tool movement is possible in one plane and the direction normal to it is used only for retracting the tool. This situation is commonly referred to as 2D milling or *pocket machining*. (Pocket refers to the region being milled).

There has been a lot of research on the subject of automatic generation of tool paths for computerized numerically controlled (CNC) pocket machining. However, much of this study, both theoretical and practical, has focused on machining pockets using a single tool; the question of how to machine pockets efficiently using more than one tool has been largely ignored, and seems to be considerably deeper and richer than the single-tool problem. Modern milling machines have the capability of automatically loading different milling tools of a wide range of radii. Using a larger tool when possible offers a significant advantage in terms of milling time. In this paper we propose a cost model for describing multiple-tool milling problem, and present an approximation algorithm.

Previous results. Within the computer-aided design and manufacturing community there has been a considerable amount of study of various heuristics for the automatic generation of tool paths for pocket machining. The most common general strategies are *contour-parallel* (also known as *window-pane milling*) [Bru82, HA92, Per78, Pre89, PK83, PK85, SL90], in which the tool spirals inwards from (or outwards to) the boundary of the region, and *axis-parallel milling* (also known as *zig-zag* or *staircase milling*) [Bru82, DBW93, Kra92, PGW90, W$^+$87], in which the milling tool moves back and forth cutting parallel strips. Multiple tool milling has been considered, see for example [BC91], but there is no theoretical analysis of the performance of the heuristics proposed.

Held [Hel89, Hel91a, Hel91b] made a comprehensive study of milling heuristics from a computational geometry perspective. For the single tool case, he presented efficient algorithms to find a feasible tool path, given the shape of the pocket to be milled and the size of the tool. On the theoretical side, Arkin, et al. [AFM93, AFM95] and Iwano, et al. [IRT94] have given constant-factor approximation algorithms for finding shortest paths for the single-tool milling problem and for the closely related problem of lawnmowing. Arkin, et al. [AHS96] have also given approximation algorithms for minimizing the number of retractions for the zig-zag milling problem, subject to the constraint that one is not allowed to mill the same region again. The problem is known to be NP-hard even when restricted to the case of a single tool [AFM93]. We know of no theoretical work considering the use of multiple tools in milling.

Domain and tools. We model the pocket machining problem as follows. Tools are changed at a designated location called the *tool-change center*. Thus the path for each tool is assumed to start and end from this center. The input to our problem provides a planar domain P to be milled, a set of tools of different sizes, and the location of the tool-change center. We make the realistic assumptions that the ratio between consecutive tool sizes is bounded (for simplicity, we assume this ratio to be bounded by 2), and that the smallest tool can mill P without the need to be lifted. (In fact, our analysis can be generalized when the assumptions do not hold, increasing the approximation ratio by at most a logarithmic factor). The tools are disks with different radii and the domain P is a connected region (possibly with holes) bounded by straight line segments and circular arcs. We call these segments and arcs *domain edges*. The tools can be moved arbitrarily. The input size n refers to the number of vertices in the domain and the parameter k refers to the number of different tools given.

Cost model. A milling plan is a sequence of tours, each for a particular tool size. In a tour, a tool alternates between being engaged with the material (milling path) or being retracted (transport path in air). Due to stress on the tool, the speed with which the tool can be moved, called the *feed rate* is typically much smaller for milling than for transport. Consider a milling plan Ψ. Define $mill(\Psi)$ to be the total milling path length for Ψ, $transport(\Psi)$ to be the total transport path length for Ψ, and $ntools(\Psi)$ to be the number of tools used by Ψ. The total milling cost in this model is $cost(\Psi) = \alpha \cdot mill(\Psi) + \beta \cdot transport(\Psi) + \gamma \cdot ntools(\Psi)$, where α, β, and γ are arbitrary nonnegative values supplied by the user as part of the input. Each milling path with tool of size t includes a cost of $2t$ in addition to the length of the path. This additional component is included to account for the time to place the cutting tool within the material. This might be done either by milling in from the side or by drilling a hole and milling down into the material. This assumption is added to prevent ridiculous solutions based on using the tool like a cookie-cutter to stamp out disks without paying any milling cost at all. From a practical standpoint, plunging the tool into material induces considerable stresses on the milling tool, and is not used in practice, or only after the time has been spent to drill a hole where the center of the milling tool is to be placed.

Output representation. As observed in [AFM95], the milling path of a tool may require a combinatorially very large description even if the size of the region milled is combinatorially very small, e.g. a small tool milling a large circle. Following the approach in [AFM95], we use a succinct representation of milling paths instead. In our output, we represent the points milled by each tool as a collection of simple regions of regular structure. If desired, the actual milling paths can be extracted by contour-parallel milling or zig-zagging within each output regions. We also output the cost of our approximate milling plan.

Outline of paper. Section 2 gives an overview of our approximation algorithms and states the main result. In Section 3 and 4, we show how to discretize the problem. Section 5 shows that our discretization method indeed approximates the milling problem to within a constant factor. Section 6 shows how to reduce the milling problem to a weighted set cover problem and describes the approximation algorithm.

2 Overview and summary of results

Milling the domain is equivalent to covering the points in the domain with copies of the tools available. Each copy of a tool used will incur some cost. This suggests that milling is related to the discrete optimization problem of weighted set-cover. Indeed, we will transform our problem to a weighted set-cover problem and then solve the weighted set-cover problem using a greedy strategy to obtain a logarithmic ratio. To construct the transformation, we need to define the elements in the base set and the weighted subsets. The transformation is not straightforward for two reasons. First, it is infeasible to use points as set elements directly as there is an infinite number of them. We overcome this by discretizing the domain into simple regions and use these simple regions as set elements instead. A nice property is that we can assume that each simple region is milled with only one tool, while increasing the approximation ratio only by a constant. Second, to produce all the weighted subsets explicitly, we need to enumerate an exponential number of combinations of these simple regions and the tools used to mill them. This is too time-consuming. We overcome this by using an approximate greedy strategy that does not require the weighted subsets to be explicitly provided. Such an approximate greedy strategy will incur another constant factor in the approximation ratio and it is based on the Euclidean k-TSP problem [Aro97]. We will be bounding the running time, space, and approximation factor in terms of k and the size of the discretization of the domain.

We first compute an $O(n)$ set of line segments for subdividing P into monotonous regions [Hel91b]. The endpoints of these line segments lie on the boundaries of P and we introduce them as extra vertices. This will split some domain edges of P into several pieces. Let P^* denote the modified domain. The size of our discretization is equal to the simple cover complexity of P^*. The *simple cover complexity*, or $scc(P^*)$, is an intrinsic measure of the geometric complexity of P^* [MMS94]. It is defined as follows. A disk is *simple* if it intersects at most 2 edges of P^*. Given any $\epsilon > 0$, we say that a ball of radius r is *ϵ-strongly simple* if the ball with the same center and radius $(1+\epsilon)r$ is simple. Given ϵ, a *strongly simple cover* of a region P^* is a collection of ϵ-strongly simple balls whose union contains P^*. Given any fixed ϵ (for example $\epsilon = 1/2$), the simple cover complexity of P^* is defined to be the cardinality of the smallest strongly simple cover of P^*. Our main result is:

Theorem 2.1 *Given a domain P of n vertices, and k circular tools, let $N = scc(P^*)$ denote the simple cover complexity of P^*. Let α, β, and γ, be nonnegative cost parameters (where $\beta < \alpha$). Then an $O(\log(kN))$-factor approximation to the optimum cost milling plan for P can be computed in time that is polynomial in n, N and k. (Constant factors hidden by the "big-Oh" do not depend on any of the input parameters.)*

One can also define a rectilinear version of the problem in which tools are squares and the domain is rectilinear. For this version, we obtain an approximation algorithm by applying similar ideas to a modification of the quadtree-based subdivision described by Mitchell, Mount, and Suri [MMS94]. The running time is polynomial in n and k. Details are omitted in this version.

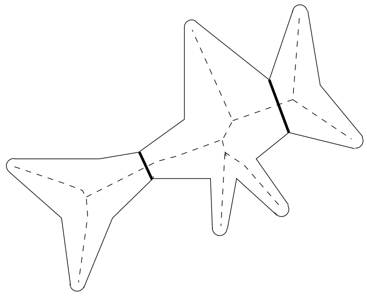

Figure 1: The dashed curves show the Voronoi diagram for the domain. The bold lines are the two cuts.

3 Subdividing the domain

Consider the Voronoi diagram for P. We define a distance function $\alpha(p)$ to ∂P from every point p on the Voronoi diagram. For each point p on the Voronoi diagram such that $\alpha(p)$ is locally minimal, we call the two nearest points on ∂P to p *bottleneck points*. We call the line segment joining them the *bottleneck segment*. We introduce all the bottleneck points to P. Each bottleneck point splits a domain edge into two smaller domain edges. We denote by P^* the resulting domain. Figure 1 shows the bottleneck segments for a simple domain.

We perform a quadtree decomposition of P^*. For any box x, we denote its side length by *width*(x). We denote by cx a box with the same center as x whose side length is $c \cdot$ *width*(x).

The goal of the decomposition is to cover P^* with a set X of boxes such that the scene within each box is extremely simple. X is generated as follows. We first enclose P^* in its bounding box. Then we apply the following splitting rule. For any box x, we call $3x$ the *buffer zone* of x and denote it by *buf*(x). Take any box x, if *buf*(x) intersects more than two domain edges, then split x through its center into four identical boxes each of half the size. When the splitting process stops (which will occur eventually since each vertex is adjacent to at most two domain edges), we obtain a set of boxes covering P^*.

A *cell* is a connected component of $P^* \cap x$ for some box $x \in X$. Given a cell C, we denote by *box*(C) the box $x \in X$ that contains C. We define the *size* of C, *size*(C), to be *width*(*box*(C)). The size of our subdivision of P^* is bounded by the *simple cover complexity* [MMS94].

Lemma 3.1 *There are $O(scc(P^*))$ boxes covering P^*. Each box x has a constant number of neighboring boxes and each neighbor has width $\Omega(width(x))$.*

4 Basic milling actions

Let $d(p, r)$ denote a disk of radius r centered at a point p. Given a tool t, we also use t to denote its radius. Whenever we put the center of t at a point p, we call

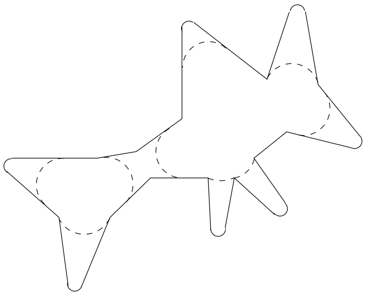

Figure 2: The accessibility arcs for a particular tool size are shown in dashed arcs.

$d(p,t)$ a *placement of* t. A placement of t is *free* if it stays within P^*. The *free space* of t is the locus of centers of all free placements of t. We denote it by $\mathcal{F}(t)$ and formally $\mathcal{F}(t) = P^* \ominus t$, where $\ominus$ is the Minkowski difference operator. Thus, $\mathcal{F}(t) \oplus t$ is the set of points in P^* that can be covered by a free placement of t, where $\oplus$ is the Minkowski sum operator.

At each vertex p of $\mathcal{F}(t)$, $d(p,t)$ touches ∂P^* at several points. If the arc α between two consecutive contact points on the boundary of $d(p,t)$ is less than a semicircle, then α belongs to the boundary of $\mathcal{F}(t) \oplus t$ and we call it an *accessibility arc*. The collection of accessibility arcs for a tool t is denoted by $\mathcal{A}(t)$. See Figure 2.

A subset K of $\mathcal{F}(t)$ and a subset $M \subseteq K \oplus t$ define a *milling action*. Specifically, t moves with center in K to remove all points in M. If K is not connected, then we have to pick t up and this will incur a charge of $O(t)$ per connected component in K plus the transport cost to visit all components in K. We say that a point in M is *milled* by this action. Note that, physically speaking, a larger set than M may be removed by t's movement. However, we only consider points in M as milled and we will have to deploy other milling actions to mill points not in M. Imagine that an arbitrary sequence of milling actions (possibly of different tools) has been applied to the collection of cells defined in the previous section. Given a cell C, we call the set of points in C not milled the *unmilled region*, and we call a connected component of the unmilled region an *unmilled component*.

Given a tool t, we restrict ourselves to two kinds of milling actions of t on a cell C. We say that t is *large* for a cell C if $t > size(C)/4$ and *small* otherwise. Conversely, we say that a cell C is small for t if $t > size(C)/4$ and large otherwise.

4.1 Large tool basic milling action

Overlay a square grid $\mathcal{G}_t$ of side length δt on P^*, where δ is any positive constant. Define $\mathcal{B}_t$ to be the set of grid squares b such that $b \oplus 4t$ overlaps some quadtree box of width less than $4t$. Any grid square $b \in \mathcal{B}_t$ induces a (possibly empty) set of milling actions as follows. If t is the smallest tool, let K be a connected component of $(b \oplus 11t) \cap \mathcal{F}(t)$; otherwise, let K be a connected component of $(b \oplus 11t) \cap \mathcal{F}(t)$ that contains some component of $b \cap \mathcal{F}(2t)$. Define $\mathcal{C}_b$ to be the collection of cells C such that t is large for C and C overlaps $b \oplus 4t$. Define M to be $K \oplus t \cap \bigcup_{C \in \mathcal{C}_b} C$. Then K and M define a *large tool basic milling action* denoted by *large_tool*(t, K, M), i.e., move t along the surface with center in K to mill points in M. We shall abbreviate basic milling action as BMA.

Lemma 4.1 *Let C be a cell small for tool t. For any large_tool(t, K, M), either $M \cap C$ is empty or the boundary of $M \cap C$ consists of points in ∂C and $\mathcal{A}(t)$.*

The above result shows that after a large tool BMA by t, new boundaries of any resulting unmilled component are accessibility arcs of t. Now, we prove that the number of large tool BMAs by all tools is polynomial in N and k. Also, the cost of a large tool BMA for tool t is $O(t)$.

Lemma 4.2 *There are $O(1)$ large tool BMAs by t induced by any grid square $b \in \mathcal{B}_t$. The cost of each such BMA is $O(t)$.*

Proof (Sketch) If t is the smallest tool, let $\mathcal{K}$ be the set of connected components of $b \oplus 11t \cap \mathcal{F}(t)$; otherwise let $\mathcal{K}$ be the set of connected components of $b \oplus 11t \cap \mathcal{F}(t)$ that contain some component of $b \cap \mathcal{F}(2t)$. Since one large tool BMA for tool t is defined for each component in $\mathcal{K}$, we need to show that the number of components in $\mathcal{K}$ is $O(1)$.

Consider the case when t is not the smallest tool. Let K be any component in $\mathcal{K}$. Since K contains a component of $b \cap \mathcal{F}(2t)$, there is a free placement $d(p, 2t)$ where $p \in K \cap b$. Clearly $d(p,t) \subseteq \mathcal{F}(t)$ since a disk of radius t with center in $d(p,t)$ lies completely within $d(p, 2t)$. Further, since $p \in b$, $d(p,t) \subseteq b \oplus 11t$. It follows that $d(p,t) \subseteq K$. By a packing argument, the number of components in $\mathcal{K}$ is $O(1)$.

If t is the smallest tool, then by our assumption $\mathcal{F}(t)$ is one connected component. It is straightforward to show that this implies that the number of components in $\mathcal{K}$ is $O(1)$. Details are omitted in this version.

Placing t with center on ∂K costs $O(t)$. Then we move t along the boundary of K and then zig-zag inside K along vertical segments that are separated by a distance at least $2t$ apart. By the result in [AFM95], the cost of this is proportional to the sum of the boundary length of K and the total length of the vertical segments. The total length of the vertical segments is $O(t)$. The boundary length of K can be shown to be $O(t)$. Thus, the cost of each *large_tool*(t, K, M) is $O(t)$. $\square$

Corollary 4.1 *There are $O(kN)$ large tool BMAs for all tools.*

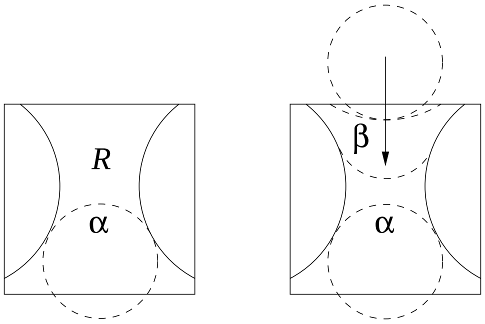

Figure 3:

Proof For each tool t, we define large tool BMAs for each $b \in \mathcal{B}_t$. Since a quadtree box of width less than $4t$ can overlap $b \oplus 4t$ for at most a constant number of grid squares b in $\mathcal{G}_t$, the number of squares in $\mathcal{B}_t$ is $O(N)$. By Lemma 4.2, the number of large tool BMAs induced by each $b \in \mathcal{B}_t$ is $O(1)$. It follows that there are $O(N)$ large tool BMAs defined for a given tool t. Summing over all tools, we have the desired result. □

4.2 Small tool basic milling action

Given a set S, we use $\mathrm{cl}(S)$ to denote its closure. Let U be an unmilled component inside C and t be small for C. Let K be a connected component of $\mathrm{cl}(U) \oplus t \cap \mathcal{F}(t)$. Define M to be $K \oplus t \cap U$. K and M define a small tool BMA by t, denoted *small_tool*(t, K, M). This action will move t on the surface with center in K to mill points in M. We show that after a small tool BMA by t, each resulting unmilled component is bounded by *exactly one* accessibility arc for t.

Lemma 4.3 *For any small_tool(t, K, M) acting on an unmilled component U, any resulting unmilled component is bounded by exactly one accessibility arc and some points in ∂C. Moreover, the accessibility arc belongs to $\mathcal{A}(t)$.*

Proof (Sketch) Let C be the cell containing U. Let *small_tool*(t, K, M) be a small tool BMA applied to an unmilled component U in C. Let W be a resulting unmilled component and so W is bounded by an accessibility arc α of radius t. Since t is small for C, α must have its endpoints on ∂P^* inside *buf*(*box*(C)). If both endpoints of α lie on one domain edge, the lemma is clearly true. Otherwise, α has endpoints on two domain edges e_1 and e_2 as shown on the left of Figure 3 and W lies in the region R shown in the figure. Assume to the contrary that W is bounded by one more accessibility arc β of some radius t'. Suppose that $t' \geq t$. Then it implies that we can advance another disk of radius t from β towards α. This disk will eventually be stopped by e_1 and e_2 as shown on the right of Figure 3. This implies that e_1 and e_2 contain a bottleneck vertex, contradiction. The case where $t' < t$ can be handled symmetrically. □

4.3 Complexity of unmilled region

In this section, we establish that after any sequence of BMAs the combinatorial complexity of an unmilled region inside a cell is always bounded by a constant. In the sequence, tools can change and large tool and small tool BMAs can interleave. The consequence is that within each cell, the number of unmilled components is always bounded by a constant and each unmilled component has constant complexity.

Lemma 4.4 *Let U be an unmilled component in a cell C after a sequence of BMAs on C. Let t be a tool small for C. After applying small_tool(t, K, M) on U, for each resulting unmilled subset W of U,*

(i) $\partial \mathrm{cl}(W)$ overlaps with some domain edge intersecting C or contains a vertex of box(C).

(ii) W has constant combinatorial complexity.

Proof If W is not bounded by ∂P^*, then for any maximal connected component s of a side of *box*(C) bounding W, s cannot be incident to two accessibility arcs as there is only one by Lemma 4.3. Thus, s is incident to a vertex of *box*(C). The complexity of W is bounded by the complexity of the arrangement of sides of *box*(C), domain edges of P^*, and accessibility arcs bounding W. There is a constant number of elements in each category and hence W has a constant complexity. □

Lemma 4.5 *After any sequence of BMAs on a cell C, the unmilled region in C has constant combinatorial complexity.*

A consequence of Lemma 4.5 is that there is a polynomial number of small tool BMAs.

Lemma 4.6 *There are $O((kN)^{O(1)})$ small tool BMAs by all tools.*

5 Approximating optimal milling using BMAs

We assume that with each BMA *large_tool*(t, K, M) or *small_tool*(t, K, M) there is an associated *starting point*, which may be any point in K. Conceptually, milling begins at the starting point and at the end the tool center is returned to this point.

The main result of this section is to show that any milling plan can be converted into a milling plan consisting entirely of BMAs while sacrificing at most a constant factor in cost.

Define $mill(t, K, M)$ to be the milling cost of the milling operation defined by the BMA $large\_tool(t, K, M)$ or $small\_tool(t, K, M)$. Given a set of BMAs S, define S_t to be the subset of S which uses tool t. Define $TSP(S)$ to be the length of a minimum Euclidean traveling salesman tour on the starting points of S and the tool change center. The cost of S is composed of two elements: the time required to perform each of its milling operations and the time to move from the starting point of one to the starting point of another. Define: $mill(S) = \sum_{(x,t,t') \in S} mill(x, t, t')$ and $transport(S) = \sum_t TSP(S_t)$.

We immediately have the following.

Lemma 5.1 *Let S be a set of BMAs which mills P^*. Then there exists a milling path Ψ, using no tools other than those used in S, such that*

$$mill(\Psi) = mill(S), \qquad transport(\Psi) = transport(S).$$

Conversely we assert that any milling plan Ψ can be transformed into a set of BMAs which mills P^*, whose milling and moving times are comparable to those of Ψ.

Theorem 5.1 *Let Ψ be a milling plan. Then there exists a set S of BMAs which mills P^* using at most a constant factor more tools as those used in Ψ, such that $mill(S) = O(mill(\Psi))$ and $transport(S) = O(mill(\Psi) + transport(\Psi))$.*

Proof (Sketch)

We show how to find a set S of BMAs corresponding to Ψ, and show how to charge each BMA to some portion of Ψ.

Large tool BMAs:

For each tool t, let Ψ_t denote the portion of the milling path only involving tool t. We think of Ψ_t as the path along which the center of the tool moves. For each continuous curve η_t in Ψ_t, we find a set of large tool BMAs using t', where t' is the tool just smaller than the smallest tool in the range $[t/2, t]$. If there is no such tool, we let t' be the smallest tool. Since successive tools differ in size by a factor of at most 2, either t' is not the smallest tool and lies between $t/4$ and $t/2$ or it is the smallest tool and lies between $t/4$ and t.

Let $\mathcal{X}_{t'} \subseteq \mathcal{B}_{t'}$ be the set of squares through which η_t passes. For each $b \in \mathcal{X}_{t'}$, we add to S all the large tool BMAs $large\_tool(t', K, M)$ induced by b such that K contains a point on η_t inside b (i.e. $K \cap \eta_t \cap b$ is not empty).

We claim the following:

(1) The total milling cost of all the large tool BMAs is no more than $O(mill(\Psi))$.

(2) Let q be a point in some cell C which is milled by Ψ using a tool of size greater than $size(C)$. Then q is milled by some large tool BMA in S.

Let L be the length of η_t. It is straightforward to show that η_t passes through at most $O(L/t' + 1)$ squares in $\mathcal{G}_{t'}$. Thus $|\mathcal{X}_{t'}|$ is $O(L/t' + 1)$. By Lemma 4.2, the number of BMAs $large\_tool(t', K, M)$ induced by any grid square $b \in \mathcal{X}_{t'}$ is bounded by a constant, and thus the total number of BMAs $large\_tool(t', K, M)$ generated by η_t is $O(L/t' + 1)$. From Lemma 4.2, we know that the milling cost of a $large\_tool(t', K, M)$ is $O(t')$, which implies that the total milling cost of all these BMAs is $O(L + t')$. This is within a constant factor of the total of the length of η_t and the placement cost of $2t$. This implies (1).

To prove (2), let $q \in C$ be milled in Ψ by t centered at point p on η_t. Let $b \in \mathcal{G}_{t'}$ be the grid square that contains p. We will show that $b \in \mathcal{B}_{t'}$, and there is a BMA $large\_tool(t', K, M)$ induced by b that is added to S and which mills q.

Since $p \in b$ and $d(p, t)$ overlaps C and $t < 4t'$, this implies that $b \oplus 4t'$ overlaps C. Also, since $size(C) < t$ and $t < 4t'$, $size(C) < 4t'$. Thus, by definition, $b \in \mathcal{B}_{t'}$, and b is used to define large tool BMAs for t'.

Let K be the connected component of $b \oplus 11t \cap \mathcal{F}(t')$ that contains point p. We claim that $large\_tool(t', K, M)$ is a valid BMA, is added to S, and it mills point q.

If t is the smallest tool, then $large\_tool(t', K, M)$ is a valid BMA. Otherwise, if t is not the smallest tool then, since $p \in \mathcal{F}(t)$ and $t > 2t'$, $p \in \mathcal{F}(2t')$. Thus K contains a component of $b \cap \mathcal{F}(2t')$ and, therefore, $large\_tool(t', K, M)$ is a valid BMA.

Since $K \cap b$ contains point p, $large\_tool(t', K, M)$ must be added to set S. To see that $large\_tool(t', K, M)$ mills point q, observe first that there is a placement $d(p', t')$ which touches point q and is contained within $d(p, t)$, and second, $p' \in K$ since we can move a disk of radius t' along the line segment from p to p', while it remains entirely within $d(p, t)$.

Call this Step 1. This puts all the large tool BMAs needed to S.

Small tool BMAs:

Next we show how to find a sequence of BMAs corresponding to Ψ for the small tools that mill some cell. This part of the proof involves a complex charging argument, and due to space limitations we only sketch the argument. Let C be a cell. Let us assume that all the large tool BMAs have already been added by the argument given above.

At this time we assert that for any point $q \in C$, if q is milled by Ψ using a tool t whose size is greater than $size(C)$, then q is milled by some large tool BMA by tool t' (where t' lies between $t/4$ and t) already added by the construction given above. Let U be any unmilled component within cell C. Any point in U can only be milled by Ψ using tool t of size less than $size(C)$, otherwise it would have been handled already by a large tool BMA. From Lemma 4.5 it follows that the unmilled region within C consists of a constant number of components, and each component is bounded by edges of the following two types:

(a) accessibility arcs generated by large tool BMAs (and later by small tool BMAs)

(b) unmilled portions of the cell boundary, including domain edges.

To account for the cost for the small tool BMAs, it will be helpful to endow each of the types of boundary elements along the unmilled component with a *charge*, proportional to its length. We show that the sum of these charges over all cells is proportional to the cost of Ψ, and then we show that the cost of the small tool BMAs for cell C is proportional to the sum of these charges for cell C. We can justify these charges as follows.

(a) The length of the accessibility arcs generated by a large tool BMA can be shown to be at most proportional to the radius of the tool. Thus we can account for these just as we accounted for the milling cost of the large tool BMA.

(b) In order to mill a portion of the boundary of C requires a milling path whose length is at least proportional to the length of the milled boundary. We charge this portion of the boundary to this milling action in Ψ. Since this is being milled with a tool that is small for C, since each cell contains a constant number of unmilled components, and since the subdivision is smooth, it follows that any segment of Ψ can be charged at most a constant number of times.

Now we proceed with the accounting for the small tool BMAs. Let U be an unmilled component inside cell C after all the large tool BMAs have acted upon it. Let T denote the set of tools that Ψ uses to mill any point in U. We add the small tool BMAs from T in decreasing order of tool size, updating the unmilled region U with each addition. Let t denote the largest of the remaining tools in T. We add the small-tool BMAs defined by U and t to set S. That is, for each component K of $U \oplus t \cap \mathcal{F}(t)$, we add a BMA $\mathit{small\_tool}(t, K, M)$. It is easy to show that there are only a constant number of such BMAs.

To account for the placement costs for these BMAs, observe that Ψ either places a copy of tool t in the vicinity, or has brought t from somewhere distant into this cell. We charge the BMA's placement against Ψ's placement or Ψ's milling cost needed to approach this cell, respectively. Again because of smoothness and the fact that the tool is small, it follows that these elements are charged a constant number of times.

To account for the milling cost of these BMAs, we make the following simple observation. Since large tools mill more efficiently than small tools, we lose nothing by making the greatest use of the largest tool that Ψ brings into an unmilled region. To make this more formal, consider the introduction of the BMA for a tool of size t, after all larger tool BMAs have already been added. Let η denote the intersection of all the milling paths in Ψ (necessarily of this and smaller tools) with K, where K is a component of $U \oplus t \cap \mathcal{F}(t)$. We charge the BMA for tool t against the length of η. We claim that the disconnected pieces of η can be easily "stitched together" using pieces from the boundary of K to get an Eulerian tour and combining this with a tour of the boundary of K, we obtain a new tour η' for t. Any point in U that can be milled by the BMA $\mathit{small\_tool}(t, K, M)$, can be milled by moving tool t along η'. The length of the BMA $\mathit{small\_tool}(t, K, M)$ is no more than the length of tour η'.

Thus, we only need to account for the length of tour η'. Observe that η' is composed of pieces from η and from the boundary of K. The pieces of η are charged against Ψ. To account for the length of the boundary of K, we need to distinguish between two types of segments on the boundary of K: (1) the boundary of $\mathcal{F}(t)$ and (2) the boundary of $U \oplus t$. Segments of type (1) are proportional in length to portions of the domain edges that get milled, and hence can be accounted for by using up the charge on this portion of the domain edge. Segments of type (2) are proportional in length an accessibility arc on the boundary of U and the cell boundary, and hence can be accounted for by using up the charge on the accessibility arc and cell boundary.

Next, we argue that accessibility arcs continue to carry charge even if after small tool BMAs have acted upon cell C. This is easy because a small tool BMA generates just one accessibility arc whose charge (which will be used in the future) can be accounted for by Ψ's placement of tool t in the vicinity of C.

Finally, we show that $\mathit{transport}(S) = O(\mathit{mill}(\Psi) + \mathit{transport}(\Psi))$. Let S_1 and S_2 be the set of large-tool and small-tool BMAs corresponding to the milling paths of tool t in Ψ. By our construction, the large-tool BMAs in S_1 use tool t', where t' lies between $t/4$ and t, and the small-tool BMAs in S_2 use tool t. Again, by our construction, for each such BMA, Ψ places a tool copy of size t that overlaps the region covered. Therefore, the transport for tool t in Ψ is as long as a TSP tour visiting the covered regions. S_1 and S_2 bring tool t' and tool t, respectively, on a tour to visit a subset of the same covered regions, but possibly at different points. Since two points in a covered region can be connected by a local milling path, we see that $\mathit{TSP}(S_1) = O(\mathit{mill}(S_1) + \mathit{transport}(\Psi_t))$ and $\mathit{TSP}(S_2) = O(\mathit{mill}(S_1) + \mathit{transport}(\Psi_t))$. Summing over all tools, we conclude that $transport(S) = O(mill(S) + transport(\Psi)) = O(mill(\Psi) + transport(\Psi))$ as $mill(S) = O(mill(\Psi))$. This completes the proof.

□

6 Greedy Approximation

We reduce the problem of finding an optimal milling plan to a *weighted set cover problem*: given a set S and a family $\mathcal{Z}$ of some subsets of S where each $X \in \mathcal{Z}$ is associated with weight $w(X)$, the objective is to find $\mathcal{Y} \subseteq \mathcal{Z}$ such that $S = \bigcup_{X \in \mathcal{Y}} X$ and $\sum_{X \in \mathcal{Y}} w(X)$ is minimized. Denote such an instance by $(S, \mathcal{Z})$. We approximate the weighted set cover problem with a greedy algorithm and thus achieving the claimed logarithmic factor approximation. The difficulty is that, as we will see, the instance of our weighted set cover problem is too large to be described explicitly. Therefore, we run an approximate greedy algorithm on a succinct problem description. We will show that this only adds an extra constant factor.

6.1 Reduction

We refine the subdivision of P^* by overlaying the accessibility arcs of $\mathcal{F}(t)$ for all t on it. The set of faces in the refined subdivision is the set S in our weighted set cover instance. It remains to define $\mathcal{Z}$.

For each $large\_tool(t, K, M)$ or $small\_tool(t, K, M)$, M is a subset of faces in S. We call K a *t-locus* and M a *t-region*. A *t-subset* is union of some t-regions and so a t-subset is a set of faces in S. For each t-subset X, we denote by X_M the set of t-regions forming X and we denote by X_K the collection of corresponding t-loci. The weight $w(X)$ of X is the sum of three terms. The first term is the milling costs of t-regions in X_M. The second term is the length of the MST connecting the tool change center and the t-loci in X_K. The third term is the sum of placement costs $O(|X_K| \cdot t)$. These terms give the total cost of milling faces in X using the BMAs induced by X_K.

We first include in $\mathcal{Z}$ the collection of all possible t-subsets for each tool t. Then we prune $\mathcal{Z}$ such that for each tool t, $X \in \mathcal{Z}$, and any cell C, if there are $small\_tool(t, K, M)$ and $small\_tool(t, K', M')$ where $K, K' \in X_K$, $M, M' \in X_M$, and $M, M' \subseteq C$, then $small\_tool(t, K, M)$ and $small\_tool(t, K', M')$ act on unmilled components of the same unmilled region of C. By our proof of Theorem 5.1, solving $(S, \mathcal{Z})$ will return at least a constant factor approximation of the milling problem.

6.2 Approximate greedy algorithm

If $(S, \mathcal{Z})$ is described explicitly, then we can use the following greedy heuristic to obtain an $O(\log kN)$ approximation factor. Initialize the cover $\mathcal{Y}$ to be empty. Compute the t-subset $X \in \mathcal{Z}$ that minimizes the *average weight* defined to be the ratio $w(X)$ divided by the number of faces in X that are currently in S. Then include X in $\mathcal{Y}$ and remove all the faces in S contained in X. Repeat until S becomes empty. This achieves an approximation factor of $\ln |S| = O(\log kN)$. Unfortunately, it is impossible to describe $(S, \mathcal{Z})$ explicitly as $|\mathcal{Z}|$ is exponential in the number of faces in Z. Instead we only store the set of t-regions for all tools and compute a t-subset of approximately minimum average weight (within a constant factor) by solving a series of instances of a variant of j-TSP. Hence, we still solve $(S, \mathcal{Z})$ within a logarithmic approximation factor.

The m-WTSP. We briefly review the j-TSP problem and introduce a variant that will be solved repeatedly as a subproblem. Given l points in the plane, the j-TSP is to find a tour of minimum length that visits any j of the l given points. The j-TSP problem is NP-hard but it can be approximated to within any constant factor in polynomial time [Aro96, Mit96]. All of the algorithms for j-TSP are based on showing that the approximate tour has a particular structure. Then the optimum tour of the particular form can be computed using dynamic programming. We generalize the j-TSP to find a tour that collects coins at the visited points. Each point p is given a table $table(p)$ and each entry of $table(p)$ is a coins-cost pair which tells the cost of collecting the associated number of coins at p. The m-WTSP is to find a tour that collects m coins so that the tour length plus the sum of the cost of collecting coins at visited points is minimized. This is defined to be the weight of the tour. The dynamic programming paradigm for approximating the j-TSP is powerful enough to approximate the m-WTSP within a constant factor in polynomial time.

Strategy and difficulty. Our strategy is to compute, for each tool t, a t-subset of approximately minimum average weight, and then return the one of the least average weight. Consider computing this t-subset for tool t. Given a t-region, fix a point p in the corresponding t-locus. Let f_p be the number of faces covered by M. Let c_p be the sum of milling cost and placement cost for the BMA by t that mills M. Create the table $table(p)$ which contains only one entry, namely, (f_p, c_p). Repeat this for all other t-regions. This yields a collection of points and their associated tables. Let F_t be the number of faces currently in S that are covered by some t-regions. For $1 \le m \le F_t$, find the approximate m-WTSP and compute its average weight. Afterwards, select the tour with minimum average weight and this corresponds to the t-subset with approximately minimum average weight. The flaw in the above strategy is that the number of faces covered when visiting a set of points is not simply the sum of numbers of faces covered when visiting each point. This is due to possible overlap among t-regions. We describe below two transformations to get around this problem and obtain the desired series of instances of m-WTSP.

Collapsing small tool BMAs. Let $large\_cell(t)$ denote the set of cells for which t is small. Given a cell $C \in large\_cell(t)$, all the t-regions inside C are generated by small tool BMAs by t. If $2box(C)$ does not contain the tool change center, then we put a representative point $pt(C)$ at the center of $box(C)$ as the common point for all t-regions in C. If $2box(C)$ contains the tool change center, then we put a representative point $pt(C)$ at the tool change center. (If $box(C)$ contains two cells, then we can put these two points slightly apart at the center of $box(C)$.) Let $\mathcal{M}$ be any milling plan. We modify $\mathcal{M}$ as follows. First, if t is transported to C and M is the first t-region in C visited by t, then we first transport t to $pt(C)$ and then to M to start milling. Afterwards, if t is transported to $pt(C)$ several times, then we transport t to $pt(C)$ exactly once and then mill all the t-regions in C that t should mill before going to another cell. Let $\mathcal{M}'$ denote the modified milling plan.

Lemma 6.1 *The tour length of t in $\mathcal{M}'$ is within a constant factor of the tour length of t in $\mathcal{M}$.*

Proof (Sketch) If we can show that detouring via the common point $pt(C)$ of a cell C increases the tour length of t by a constant factor, then it is clear that we can make t being transported to $pt(C)$ exactly once. Thus, we focus on proving that the detour is not expensive. Let $\mathcal{M}^*$ denote the modified $\mathcal{M}$ with the detour. Let T and T^* be a tour of t in $\mathcal{M}$ and its modified version in $\mathcal{M}^*$ respectively. We first modify T^* as follows. In T^*, if there is an edge e from a t-locus K_1 for a t-region in cell C_1 to $pt(C_2)$ and then to the t-locus K_2 for a t-region in cell C_2, then we replace e by a path of two edges: from K_1 to K_2 and then to $pt(C_2)$. Let $\tilde{T}$ denote the modified T^* and the length of $\tilde{T}$ is no less than the length of T^* by triangle inequality. We call an edge in $\tilde{T}$ between a t-locus for a t-region in a cell C and $pt(C)$ a detour edge. By construction, $\tilde{T}$ contains all the edges in T and some detour edges. Let e be a detour edge from a point p to $pt(C)$ for some cell C. If $2box(C)$ contains the tool change center, then $pt(C)$ is the tool change center. Then we charge e to the entire T. Suppose that $2box(C)$ does not contain the tool change center. In T, after leaving the point p, the tour has to leave $2box(C)$ eventually. Let ρ be the path in T starting at p and ending at the boundary of $2box(C)$. The length of ρ is $\Omega(size(C))$. We charge the length of e in $\tilde{T}$ to line segments on ρ proportional to their lengths. By smoothness, it is not hard to see that only a constant number of boxes can have detour edges charged to ρ. The result then follows since there is a constant number of detour edges for each box. $\square$

For each cell $C \in large\_cell(t)$, we associate a table $table(pt(C))$ with $pt(C)$ to reflect all the possible effects of small tool BMAs by t in C and the corresponding cost. We enumerate all possible unmilled regions in C and for each unmilled region, we enumerate all possible combinations of small tool BMAs on unmilled components. Note that each unmilled component induces at most a constant number of small tool BMAs. For each such combination of small tool BMAs by t, we find out the number of faces covered and the cost (sum of costs of the small tool BMAs, MST length connecting $pt(C)$ and the t-loci, and the placement cost). The table $table(pt(C))$ has $O((kN)^{O(1)}2^{O(1)})$ entries.

Separating large tool BMAs. After collapsing small tool BMAs by a tool t, we have one representative point for each cell $C \in large\_cell(t)$. We now address the overlapping among t-regions of large tool BMAs. We divide such t-regions into groups so that within a group, no two t-regions overlap. The division is done by coloring an induced graph as follows. Consider the grid squares that induce the large tool BMAs by t. There is a constant number of large tool BMAs induced by each grid square. Since each grid square has width $\Omega(t)$, the t-region of a large tool BMA cannot overlap more than a constant number of grid squares. Thus, the t-region of a large tool BMA overlaps at most a constant number of t-regions of other large tool BMAs. This induces a graph of maximum degree bounded by a constant Δ. Such a graph is colorable using at most $\Delta + 1$ colors which yields at most $\Delta+1$ groups of t-regions. We fix a point p in each t-region M and associate with it a table $table(p)$ of a single entry, namely, the faces covered by M and cost (cost of the corresponding large tool BMA plus placement cost). Thus, we obtain at most $\Delta + 1$ groups of points each associated with a table. We denote these groups by $G_i(t)$ for $1 \leq i \leq \Delta + 1$. Finally, we add one last group $G_{\Delta+2}(t)$ which contains $pt(C)$ along with $table(pt(C))$ for each cell $C \in large\_cell(t)$.

Lemma 6.2 *There is a milling plan $\mathcal{A}''$ in which each tour of a tool t beginning and ending at the tool change center visits only points in some $G_i(t)$. Moreover, $\mathcal{A}''$ approximates the optimal milling plan within a constant factor.*

The complete algorithm. The following is the complete algorithm. Recall that F_t is the number of faces currently in S covered by some t-regions.

```
while S ≠ ∅ do
   for each tool t do
      for 1 ≤ m ≤ F_t do
         for each G_i(t) do
            solve the m-WTSP on G_i(t)
         end for
         T_m(t) = the minimum weight tour;
```

    end for
    $T(t)$ = minimum average weight $T_m(t)$;
  end for
  T = minimum average weight $T(t)$;
  remove faces covered in T from S;
  update F_t for all t;
  update the numbers of faces in all tables;
end while

Theorem 6.1 *An approximate milling plan can be computed in $O((kN)^{O(1)})$ time with cost within $O(\log kN)$ of the optimal.*

Acknowledgments

We thank Ajay Joneja of the Industrial Engineering Department at the Hong Kong University of Science and Technology for suggesting the multiple-tool milling problem and for valuable discussions on the practical aspects of milling.

References

[AFM93] E. Arkin, S. Fekete, and J. Mitchell. The lawnmower problem. In *Proc. 5th Canad. Conf. Comput. Geom.*, pages 461–466, 1993.

[AFM95] E. Arkin, S. Fekete, and J. Mitchell. Approximation algorithms for lawnmowing and milling. Report, SUNY Stony Brook, 1995.

[AHS96] Esther M. Arkin, Martin Held, and Christopher L. Smith. Optimization problems related to zigzag pocket machining. In *Proc. 7th ACM-SIAM Sympos. Discrete Algorithms*, pages 419–428, 1996.

[Aro96] S. Arora. Polynomial time approximation schemes for Euclidean TSP and other geometric problems. In *Proc. 37th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 2–11, 1996.

[Aro97] S. Arora. Nearly linear time approximation schemes for Euclidean TSP and other geometric problems. In *Proc. 38th Annu. IEEE Sympos. Found. Comput. Sci.*, 1997.

[BC91] M. Bala and T.-C. Chang. Automatic cutter selection and optimal cutter path generation for prismatic parts. *Int. J. Prod. Res*, 29:2163–2176, 1991.

[Bru82] L. K. Bruckner. Geometric algorithms for 2.5d roughing process of sculptured surfaces. In *Proc. Joint Anglo-Hungarian Seminar on Computer-Aided Geometric Design*, 1982.

[DBW93] A. V. Deshmukh, M. M. Barash, and H. P. Wang. On selection of tool path orientations for generating prismatic features. Report, IE Dept, Purdue U, 1993.

[HA92] A. Hansen and F. Arbab. An algorithm for generating NC tool paths for arbitrarily shaped pockets with islands. *ACM Trans. Graph.*, 11(2):152–182, April 1992.

[Hel89] M. Held. Geopocket – A sophisticated computational geometry solution of geometrical and technological problems arising from pocket machining. In F. Kimura and A. Rolstadås, editors, *Proc. Computer Applications in Production and Engineering*, pages 283–293. North-Holland, October 1989.

[Hel91a] M. Held. A geometry-based investigation of the tool path generation for zigzag pocket machining. *Visual Comput.*, 7(5–6):296–308, September 1991.

[Hel91b] M. Held. *On the Computational Geometry of Pocket Machining*, volume 500 of *Lecture Notes Comput. Sci.* Springer-Verlag, June 1991.

[IRT94] K. Iwano, P. Raghavan, and H. Tamaki. The traveling cameraman problem, with applications to automatic optical inspection. In *Proc. 5th Annu. Internat. Sympos. Algorithms Comput.*, Lecture Notes Comput. Sci., Beijing, 1994. Springer-Verlag.

[Kra92] T. R. Kramer. Pocket milling with tool engagement detection. *J. of Maufacturing Systems*, 11(2):114–123, 1992.

[Mit96] Joseph S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple new method for the geometric k-MST problem. In *Proc. 7th ACM-SIAM Sympos. Discrete Algorithms*, pages 402–408, 1996.

[MMS94] J. S. B. Mitchell, D. M. Mount, and S. Suri. Query-sensitive ray shooting. In *Proc. 10th Annu. ACM Sympos. Comput. Geom.*, pages 359–368, 1994.

[Per78] H. Persson. NC machining of arbitrarily shaped pockets. *Comput. Aided Design*, 10(3):169–174, May 1978.

[PGW90] P. V. Prabhu, A. K. Gramopadhye, and H. P. Wang. General mathematical model to optimizing NC tool path for face milling of flat convex polygonal surfaces. *Int. J. Production Research*, 28(1):101–130, 1990.

[PK83] K. Preiss and E. Kaplansky. Automatic mill routing from solid geometry information. In *Proc. Computer Applications in Production and Engineering*. North-Holland, 1983.

[PK85] K. Preiss and E. Kaplansky. Automated CNC milling by artificial intelligence methods. *J. of Maufacturing Systems*, 4(1):51–63, 1985.

[Pre89] K. Preiss. Automated mill pocketing computations. In *Advanced Geometric Modeling for Engineering Applications*. North-Holland, 1989.

[SL90] Y. S. Suh and K. Lee. NC milling tool path generation for arbitrary pockets defined by sculptured surfaces. *Comput. Aided Design*, 22(5):273–284, June 1990.

[W+87] H.-P. Wang et al. On the efficiency of NC tool path planning for face milling operations. *Trans. of the ASME, J. of Engineering for Industry*, 109(4):370–376, November 1987.