Cutting a Cornered Convex Polygon Out of a Circle

Syed Ishtiaque Ahmed, Md. Ariful Islam and Masud Hasan
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology
Dhaka 1000, Bangladesh
Email: ishtiaque@csebuet.org, arifulislam@csebuet.org, masudhasan@cse.buet.ac.bd

***Abstract*— The problem of cutting a convex polygon P out of a piece of planar material Q with minimum total cutting length is a well studied problem in computational geometry. Researchers studied several variations of the problem, such as P and Q are convex or non-convex polygons and the cuts are line cuts or ray cuts. In this paper we consider yet another variation of the problem where Q is a circle and P is a convex polygon such that P is bounded by a half circle of Q and all the cuts are line cuts. We give two algorithms for solving this problem. Our first algorithm is an $O(\log n)$-approximation algorithm with $O(n)$ running time, where n is the number of edges of P. The second algorithm is a constant factor approximation algorithm with approximation ratio 6.48 and running time $O(n^3)$.**

***Index Terms*— algorithms, approximation algorithms, computational geometry, line cut, ray cut, polygon cutting, rotating calipers**

I. INTRODUCTION

The problem of cutting small polygonal objects "efficiently" out of larger planar objects arises in many industries, such as metal sheet cutting, paper cutting, furniture manufacturing, ceramic industries, fabrication, ornaments, and leather industries. This type of problems are in general known as *stock cutting problems* [1]. Different types of cuts and different criteria for efficiency of cutting are considered in cutting processes, mostly depending up on the types of the materials. The two most common types of cuts are the *line cuts* (i.e., *guillotine cuts*) and *ray cuts*. A line cut is a line that cuts the given object into several pieces and does not run through the target object. A ray cut is a ray that runs from the infinity to a certain point of the given object, possibly to a boundary point of the target object. A line cut is feasible for cutting out convex objects since it cuts an object into many pieces below and above the cut. On the other hand, ray cuts can be performed by many types of saws such as scroll saw, band saw, laser saw and wire saw [2]. Ray cuts can cut out non-convex objects too. But at the same time they need to make turns in the cutting process and so, needs some "clearance" for a turn, which can make it impossible to cut an arbitrary non-convex polygon. In particular, for applying ray cuts to a non-convex polygon it is necessary for the polygon to have no "twisted pockets", i.e., part of the polygon boundary that does not see the infinity. As a whole, a cutting process that uses only line cuts is much simpler than that uses only ray cuts.

In a cutting process the main criteria for "efficiency" of cutting is to minimize the total cutting length, which is also known as the *cutting cost*. While cutting a convex polygon it may be true that the cutting cost for ray cuts are less than that for line cuts. But due to the above simplicity line cuts are more popular for cutting convex objects and are well studied as well, at least theoretically [1], [3]–[9]. Moreover, it can be shown that [9] it is not always possible to replace line cuts by ray cuts to get better cutting cost.

In this paper we consider the problem of cutting a convex polygon P out of a circle Q by using line cuts where P is "much smaller" than Q, namely P is *cornered convex* with respect to Q. A *cornered convex polygon* P inside a circle Q is a convex polygon which is positioned completely on one side of a diameter of Q. See Fig. 1(a). The *(cutting) cost* of a line cut is the length of the intersection of the line with Q. After a cut is made, Q is updated to the piece containing P. A *cutting sequence* is a sequence of cuts such that after the last cut in the sequence we have $P = Q$. We give algorithms for cutting P out of Q by line cuts with total cutting cost of the cutting sequence as small as possible. See Fig. 1(b). In many applications, such as metal sheet cutting, it is natural to have the given object as a large circular sheet and the target object as a sufficiently smaller convex polygon.

A. Known results

If Q is another convex polygon with m edges, this problem with line cuts has been approached in various ways by many researchers in computational geometry community [1], [3]–[9]. Overmars and Welzl first introduced this problem in 1985 [3]. If the cuts are allowed only along the edges of P, they proposed an $O(n^3 + m)$-time algorithm for this problem with optimal cutting length, where n is the number of edges of P. The problem is more difficult if the cuts are more general, i.e., they are not restricted to touch only the edges of P. In that case Bhadury and Chandrasekaran showed that the problem has optimal solutions that lie in the algebraic extension of the input data field [1] and due to this algebraic nature of

doi:10.4304/jcp.5.1.4-11

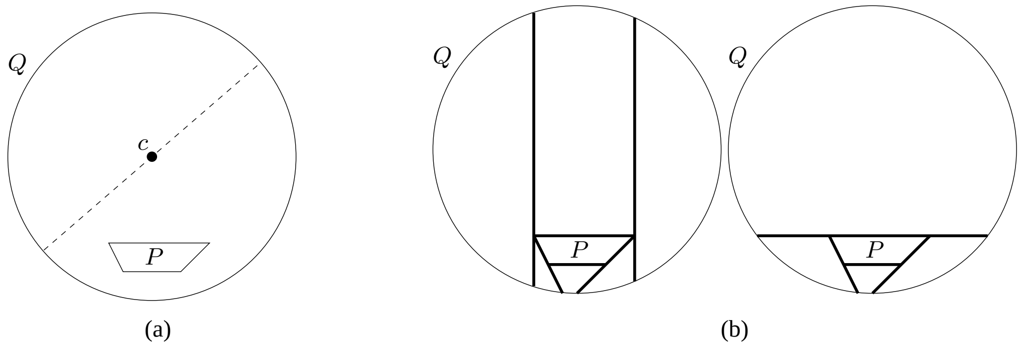


Figure 1. (a) A cornered convex polygon P inside a circle Q. (b) Two different cutting sequences (bold lines) to cut P out of Q; Cutting cost of the sequence in the left figure is more than that in the right figure.

this problem, an approximation scheme[1] is the best that one can achieve [1]. They also gave an approximation scheme with pseudo-polynomial running time [1].

After the indication of Bhadury and Chandrasekaran [1] to the hardness of the problem, several researchers have given polynomial time approximation algorithms. Dumitrescu proposed an $O(\log n)$-approximation algorithm with $O(mn+n\log n)$ running time [5], [6]. Then Daescu and Luo [7] gave the first constant factor approximation algorithm with ratio $2.5+||Q||/||P||$, where $||Q||$ and $||P||$ are the perimeter of Q and P respectively. Their algorithm has a running time of $O(n^3+(n+m)\log(n+m))$. The best known constant factor approximation algorithm is due to Tan [4] with an approximation ratio of 7.9 and running time of $O(n^3+m)$. In the same paper [4], the author also proposed an $O(\log n)$-approximation algorithm with improved running time of $O(n+m)$. As the best known result so far, very recently, Bereg, Daescu and Jiang [8] gave a polynomial time approximation scheme (PTAS) for this problem with running time $O(m+\frac{n^6}{\epsilon^{12}})$.

For ray cuts, Demaine, Demaine and Kaplan [2] gave a linear time algorithm to decide whether a given polygon P is *ray-cuttable* or not. For optimally cutting P out of Q by ray cuts, if Q is convex and if P is non-convex but ray-cuttable, then Daescu and Luo [7] gave an almost linear time $O(\log^2 n)$-approximation algorithm. If P is convex, then they gave a linear time 18-approximation algorithm. Tan [4] improved the approximation ratio for both cases as $O(\log n)$ and 6, respectively, but with much higher running time of $O(n^3+m)$.

B. Our results

All the previous results consider Q as a polygon (either convex or non-convex). However, to our knowledge, no algorithm is known when Q is a circle. In this paper, we consider the problem where Q is a circle and P is a cornered convex polygon inside Q. We give two approximation algorithms for this problem. Our first algorithm has an approximation ratio of $O(\log n)$ and runs in $O(n)$ time. Our second algorithm has an approximation ratio of 6.48 and runs in $O(n^3)$ time.

C. Comparison of the results

While for an approximation algorithm a constant factor approximation ratio is preferable to input-dependent approximation ratio, the linear running time of our first algorithm is much better than the cubic running time of our second algorithm. The ratio 6.48 of our second algorithm is better than that of 7.9 of Tan's algorithm [4] (although the later deals with Q as a convex polygon).

When both P and Q are convex polygons, almost all the existing algorithms on line cuts use two major steps: cutting a minimum area rectangle or a minimum area triangle from Q that bounds P and then cutting P out of that bounding box. Our algorithms also follow similar approach. However, we observe that the existing algorithms can not be applied directly to solve our problem. Moreover, the running time of those algorithms are too high compared to our algorithms. In particular, Tan's [4] constant factor approximation algorithm takes $O(n^3+m)$ time, the $O(\log n)$-approximation algorithm of [5], [6] takes $O(mn+n\log n)$ time and the PTAS of Bereg et.al. [8] takes $O(m+\frac{n^6}{\epsilon^{12}})$ time, where m can be arbitrarily large. In contrast, the running time of our algorithms are free of m and one of them is linear. Also observe that in the existing algorithms, increasing the value of m for "approximating" Q to a circle makes them inefficient. See TABLE I for a summary of comparisons among the existing algorithms and our algorithms.

D. Outline

The rest of the paper is organized as follows. We give some definitions and preliminaries in Section II. Then we present our two algorithms in Section III and Section IV respectively. Finally, Section V concludes the paper with some future works.

II. Preliminaries

A line cut is a *vertex cut* through a vertex v of P if it is tangent to P at v. Similarly, a line cut is an *edge*

[1]A *ρ-approximation algorithm* (similarly, an *approximation scheme*) has a cutting length that is ρ times (similarly, $(1+\epsilon)$ times, for any value $\epsilon > 0$) the optimal cutting length. Please refer to [10] for preliminaries on approximation algorithms.

Cut Type	Q	P	Approx. Ratio	Running Time	Reference
Ray cuts	-	Non-convex	Ray-cuttable?	$O(n)$	[2]
	Convex	Convex	18	$O(n)$	[7]
	Convex	Non-convex	$O(\log^2 n)$	$O(n)$	[7]
	Convex	Convex	6	$O(n^3+m)$	[4]
	Convex	Non-convex	$O(\log n)$	$O(n^3+m)$	[4]
Line cuts	Convex	Convex	$O(\log n)$	$O(mn+n\log n)$	[5], [6]
	Convex	Convex	$2.5+\|\|Q\|\|/\|\|P\|\|$	$O(n^3+(n+m)\log(n+m))$	[7]
	Convex	Convex	7.9	$O(n^3+m)$	[4]
	Convex	Convex	$(1+\epsilon)$	$O(m+\frac{n^6}{\epsilon^{12}})$	[8]
	Circle	**Cornered convex**	$\boldsymbol{O(\log n)}$	$\boldsymbol{O(n)}$	**This paper**
	Circle	**Cornered convex**	**6.48**	$\boldsymbol{O(n^3)}$	**This paper**

TABLE I.
COMPARISON OF THE RESULTS.

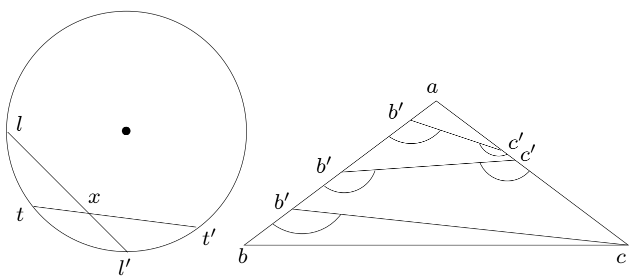
cut through an edge e of P if it contains e. At any time the edges of P through which an edge cut has passed are called *cut edges* of P and other edges of P are called *uncut edges* of P. To cut P out of Q all n edges of P must become cut edges and for that we require exactly n edge cuts. However, applying only edge cuts may not give an optimal solution and we need vertex cuts as well.

In the rest of this section we give some elementary geometry that plays important role in our paper. Let c be the center of Q. An edge e of P is *visible* from c if for every point p of e the line segment cp does not intersect P in any other point. So, if e is collinear with c, then we consider e as invisible. Similarly, a vertex v of P is *visible* from c if the line segment cv does not intersect P in any other point.

In this paper we do not consider the diameter of Q as a *chord*, i.e., a chord is always smaller than a diameter. Let ll' be a chord of Q. ll' divides Q into two circular segments, one is bigger than a half circle and the other one is smaller than a half circle. Let tt' be another chord intersecting ll' at x such that tx is in the smaller circular segment of ll'. See Fig. 2(a). The following two lemmas are obvious and their illustration can be found in Fig. 2.

Lemma 1: xt is no bigger than ll'.

Lemma 2: Let $\triangle abc$ be an obtuse triangle with the obtuse angle $\angle bac$. Consider any line segment connecting two points b' and c' on ab and ac, respectively, possibly one of b' and c' coinciding with b or c respectively. Then the angle $\angle bb'c'$ and $\angle cc'b'$ are obtuse.



(a) (b)

Figure 2. Illustration of Lemma 1 and Lemma 2.

III. ALGORITHM 1

Our first algorithm has four phases : (1) D-separation, (2) triangle separation, (3) obtuse phase and (4) curving phase. In *D-separation phase*, we cut out a small portion of Q (less than the half of Q) which contains P (and looks like a "D"). Then in *triangle separation phase* we reduce the size of Q even more by two additional cuts and bound P by almost a triangle. In *obtuse phase* we assure that all the portions of Q that are not inside P are inside some obtuse triangles. Finally, in *curving phase* we cut P out of Q by cutting those obtuse triangles in rounds.

Let C^* be the optimal cutting length to cut P from Q. Clearly C^* is at least the length of the perimeter of P.

A. D-separation

A *D* of the circle Q is a circular segment of Q which is smaller than an half circle of Q. By a *D-separation* of P from Q we mean a line cut of Q that creates a D containing P. In general, for a circle Q and a convex polygon P there may not exist any D-separation of P. But in our case since P is cornered, there always exists a D-separation of P. We first find a D-separation that has minimum cutting cost.

Lemma 3: A minimum-cost D-separation, C_1, can be found in $O(n)$ time.

Proof: Clearly C_1 must touch P. So, C_1 must be a vertex cut or an edge cut of P. Observe that any vertex cut or edge cut that is a D-separation must be through a visible vertex or a visible edge. Let e be a visible edge of P. Let ll' be a line cut through e. Let cp be the line segment perpendicular to ll' at p. If p is a point of e, then we call e a *critical edge* of P and ll' a *critical edge cut* of P. Since P is convex, it can have at most one critical edge. Similarly, let v be a vertex of P and let tt' be a vertex cut through v. Let cp be the line segment perpendicular to tt' at p. If tt' is such that $p = v$, then we call v a *critical vertex* of P and tt' a *critical vertex cut* of P. Again, P can have at most one critical vertex. Moreover, P has a critical edge if and only if it does not have a critical vertex. See Fig. 3. Now, if P has the critical edge e (and no critical vertex), then C_1 is the corresponding critical edge cut ll'. C_1 is minimum, because any other vertex cut or edge cut of P is either closer to c (and thus bigger) or does not separate c from P. On the other hand, if P has the critical vertex v, then C_1 is the corresponding critical vertex cut tt' of P. Again, C_1 is minimum by the same reason.

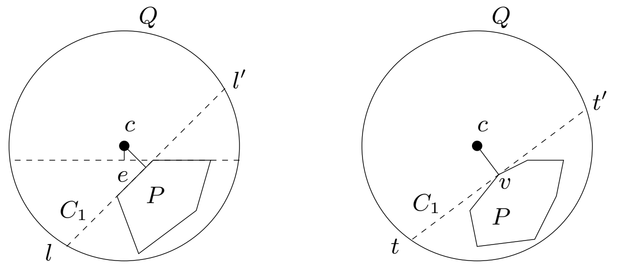


Figure 3. Critical edge and critical vertex.

For running time, all visible vertices and visible edges of P can be found in linear time. Then finding whether an edge of P is critical takes constant time. Over all edges finding the critical edge, if it exists, takes $O(n)$ time. Similarly, for each visible vertex v we can check in constant time whether v is critical or not by comparing the angles of cv with two adjacent edges of v, which takes constant time. Over all visible vertices, it takes linear time. ■

Lemma 4: Cost of C_1 is at most C^*.

Proof: Consider any optimal cutting sequence $\mathcal{C}$ with cutting cost C^*. $\mathcal{C}$ must separate P from c. However, it may do that by using a single cut (Case 1) or by using more than one cut (Case 2). In Case 1, if it uses a single cut then it is in fact doing a D-separation. By Lemma 3, it can not do better than C_1, and therefore, cost of C_1 is at most C^*.

In Case 2, there are several sub-cases. We will prove that in any case the cutting cost of separating P from c is even higher than that for a single cut. Let C be the first cut in $\mathcal{C}$ that separates c from P. C can not be the very first cut of $\mathcal{C}$, otherwise, it is doing a D-separation and we are in Case 1. It implies that C is not a (complete) chord of Q. For the rest of the proof please refer to Fig. 4. Let a and a' be the two end points of C. Let bb' be the chord of Q that contains aa' where b is closer to a than to a'. At least one of a and a' is not incident on the boundary of Q. We first assume that a is not incident to the boundary of Q but a' is (Case 1a). Let $C_x = xx'$ be the cut that was applied immediately before C and intersects C at a. Since C_x does not separate c from P, the bigger circular segment created by C_x contains P, c and a. Now if x' is an end point of C_x, then by Lemma 1 ab is smaller than xx'. It implies that having C_x in addition to C_1 increases the cost of separating P from c (see Fig. 4(a)). Similarly, if x' is not an end point of C_x, and thus another cut is involved, then by Lemma 1 the cost will be even more (see Fig. 4(b)).

Now consider the case when both a and a' are not incident on the boundary of Q (Case 1b). So at least two cuts are required to separate ab and $a'b'$ from bb'. Let those two cuts be C_x and C_y respectively. Let xx' and yy' be the two chords of Q along which C_x and C_y are applied. Now if C_x and C_y do not intersect, then for each of them by applying the argument of Case 1a we can say that the cutting cost is no better than that for a single cut. But handling the case when C_x and C_y intersect is not obvious (see Fig. 4(c, d)). Let z' be their intersection point. Again, there may be several sub-cases: x and y may or may not be end points of C_x and C_y. Assume that both x and y are end points of C_x and C_y. Remember that none of C_x and C_y separates P from c. So, the region bounded by $xz'y$ must contain P and c inside of it. In that case the total length of xz' and yz' is at least the diameter of Q, which is bigger than bb' (see Fig. 4(c)). For the other cases, where at least one of x and y is not an end point of C_x and C_y, respectively, by Case 1a the cost is even more (for example, see Fig. 4(d)). ■

B. Triangle separation

In this phase we apply two more cuts C_2 and C_3 and "bound" P inside a "triangle". From there we achieve three triangles inside which we bound the remaining uncut edges of P. (In the D-separation phase, at most one edge of P becomes cut).

Let $C_1 = aa'$ be the cut applied during the D-separation. We apply two cuts $C_2 = at$ and $C_3 = a't'$ such that both of them are also tangents to P. If C_2 and C_3 intersect (inside Q or on the boundary of Q), then let z be the intersection point (see Fig. 5(a)), otherwise let z be the point outside Q where the extensions of C_2 and C_3 intersect (see Fig. 5(b)). We get three resulting triangles $T_a, T_{a'}, T_z$ having a, a' and z, respectively, as a peak. We only describe how to get T_a; description for $T_{a'}$ and T_z are analogous. If C_1 is an edge cut, then let rr' be the corresponding edge such that a is closer to r than to r' (see Fig. 5(a)). If C_1 is a vertex cut, then let r be the corresponding vertex of P. Let s be the similar vertex due to C_2 (see Fig. 5(b)). Then $T_a = \triangle ars$. The polygonal chain of P *bounded* by T_a is the edges from r to s that reside inside T_a.

Lemma 5: Total cost of C_2 and C_3 to achieve $T_a, T_{a'}$ and T_z is at most $2C^*$. Moreover, C_2 and C_3 can be found in linear time.

Proof: Whether z is within Q or outside Q, the length of at and $a't'$ can not be more than twice the length of aa'. By Lemma 3, aa' is no more than C^*. Therefore, the total cost of C_2 and C_3 is at most $2C^*$.

To find at linearly, we can simply scan the boundary of P starting from the vertex or edge where aa' touches P and check in constant time whether a tangent of P is possible through that vertex or edge. Similarly we can find $a't'$ within the same time. ■

C. Obtuse phase

Consider the triangle $T_a = \triangle ars$ obtained in the previous phase. We call the vertex a the *peak* of T_a and the edge rs the *base* of T_a. Observe that the angle of T_a at a is acute. Similarly, the angle of $T_{a'}$ at its peak a' is

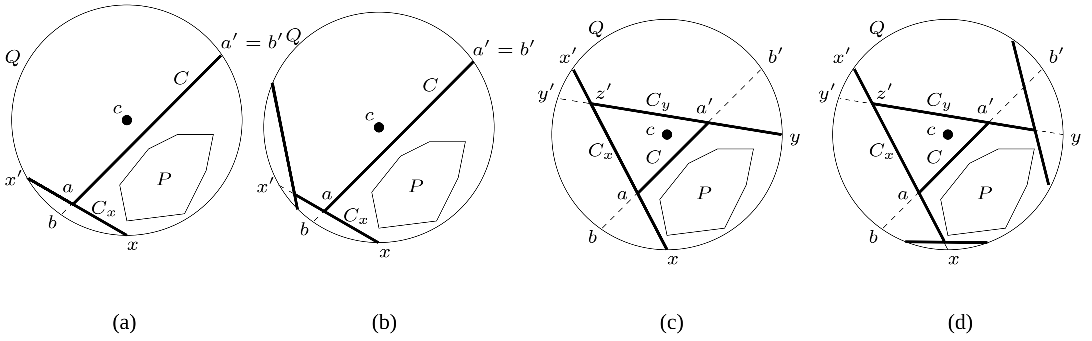


Figure 4. Separating P from c by using more than one cut. Bold lines represent the cutting cost.

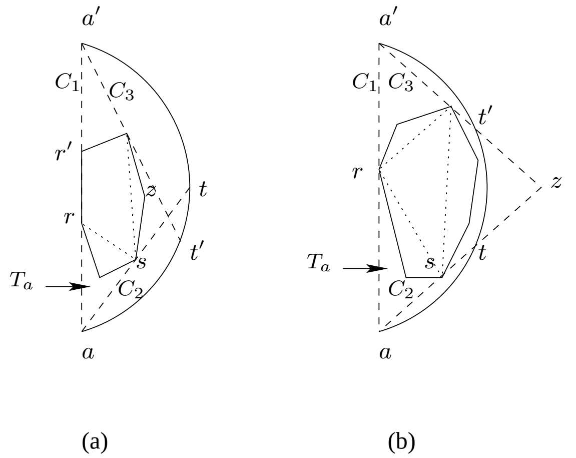


Figure 5. Triangle separation.

also acute. However, the angle of T_z at its peak z may be acute or obtuse.

For each of $T_a, T_{a'}$ and T_z (if T_z is acute) we apply a cut and obtain one or two triangles such that the angle at their peaks are obtuse and they jointly bound the polygonal chain of the corresponding triangle. We describe the construction of the triangle(s) obtained from T_a only; description for the triangles obtained from $T_{a'}$ and T_z are analogous.

Please refer to Fig. 6. Let P_a be the polygonal chain bounded by T_a. Length of ar and as may not be equal. W.l.o.g. assume that as is not larger than ar. Let s' be the point on ar such that length of as' equals to the length of as. Connect ss' if s' is different from r. We will find a line segment inside T_a such that it is tangent to P_a and is parallel to ss'. We have two cases: (i) ss' itself is a tangent to P_a, and (ii) ss' is not a tangent to P_a.

Case (i): If ss' itself is tangent to P_a (at s), we apply our cut along ss'. This cut may be a vertex cut through s or an edge cut through the edge of P_a that is incident to s. If it is a vertex cut then we get the resulting obtuse triangle $\triangle rss'$ and the polygonal chain bounded by this triangle is same as P_a (see Fig. 6(a)). On the other hand, if the cut is an edge cut, let u be the other vertex of the cut edge. Then our resulting obtuse triangle is $\triangle rus'$ and the polygonal chain bounded by this triangle is the edges from r to u (see Fig. 6(b)). Since as and as' are of same length, in either case the triangle $\triangle rss'$ or $\triangle rus'$ has obtuse angle at s'.

Case(ii): For this case, let u, u' be two points on as and ar, respectively, such that uu' is tangent of P_a and is parallel to ss'. We apply the cut along uu'. Again, this cut may be a vertex cut or an edge cut. If it is a vertex cut, let g be the vertex of the cut. Then we get two obtuse triangles $\triangle ru'g$ and $\triangle sug$ and the polygonal chains bounded by them are the sets of edges from r to g and from g to s respectively (see Fig. 6(c)). If it is an edge cut, then let gg' be the edge of the edge cut with u being closer to g than to g'. Then we get two obtuse triangles $\triangle ru'g'$ and $\triangle sug$ and the polygonal chains bounded by them are the sets of edges from r to g' and from g to s respectively (see Fig. 6(d)). Again, since au and au' are of same length, in either case the pair of triangles have obtuse angles at u and u' respectively.

Lemma 6: Total cost of obtaining obtuse triangles from $T_a, T_{a'}$, and T_z is at most C^*. Moreover, they can be found in $O(\log n)$ time.

Proof: Consider the construction of the obtuse triangle(s) from T_a. Length of the cut ss' or uu' is at most the length of rs, which is bounded by the length of P_a. Over all three triangles $T_a, T_{a'}$ and T_z, the total cutting length is bounded by the length of the perimeter of P. Since C^* is at least the length of the perimeter of P, the first part of the lemma holds.

For running time, in T_a, we can find the tangent of P_a in $O(\log |P_a|)$ time by using a binary search, where $|P_a|$ is the number of edges in P_a. Therefore, over all three triangles $T_a, T_{a'}$ and T_z, we need a total of $O(\log n)$ time. ■

D. Curving phase

After the obtuse phase the edges of P that are not yet cut are partitioned and bounded into polygonal chains with at most six obtuse triangles. In this phase we apply the cuts in rounds until all edges of P are cut. Our cutting procedure is same for all obtuse triangles and we describe for only one.

Let $T_u = \triangle gus$ with peak u and base gs be an obtuse triangle. (See Fig. 7). Let the polygonal chain bounded by T_u be P_u. Let the edges of P_u be $e_1, \ldots, e_k$ with $k \geq 2$.

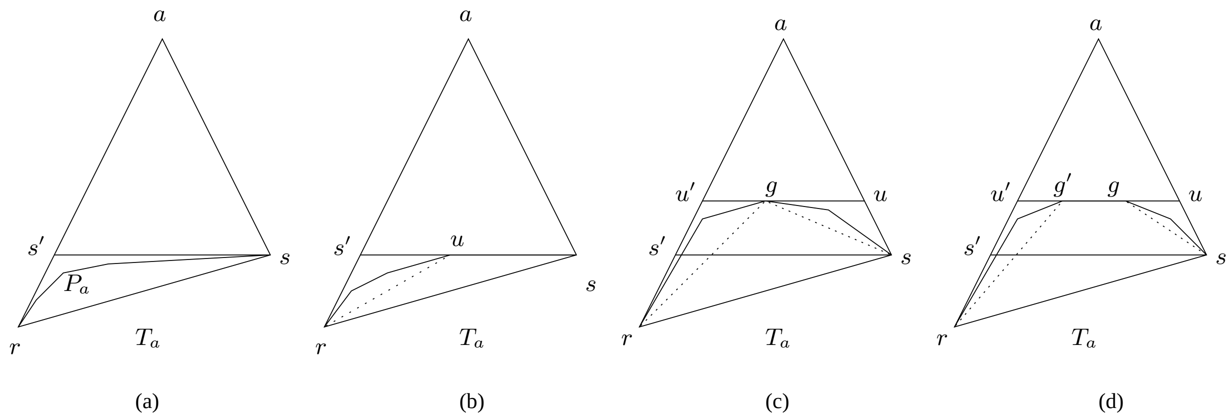


Figure 6. Obtaining obtuse triangle(s) from T_a.

We shall apply cuts in rounds and all cuts will be edge cuts. In the first round we apply an edge cut C' along the edge $e_{k/2}$. Then we connect g and s with the two end points of $e_{k/2}$ to get two disjoint triangles. Since T_u is obtuse, by Lemma 2 these two new triangles are also obtuse. In the next round we work on each of these two triangles recursively and continue until all edges of P_u are cut.

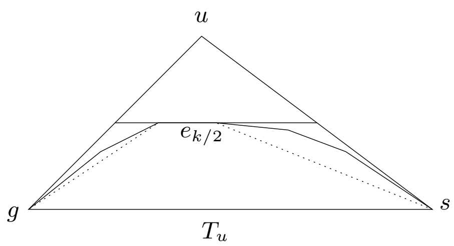


Figure 7. Curving phase

Lemma 7: In curving phase, to cut all edges of P_u there can be at most $O(\log k)$ rounds of cuts. Moreover, the total cost of these cuts is $|P_u| \log k$, where $|P_u|$ is the number of edges of P_u, and the running time is $O(|P_u|)$.

Proof: At each round the number of triangles get doubled and the number of edges that become cut also get doubled. So after $\log k$ rounds all k edges are cut.

Let the length of P_u be L_u. At each round the total length of the bases plus the length of the edges that are being cut is no more than L_u. So, the total cutting cost is $L_u \log k$ over all rounds.

For running time, finding the edge $e_{k/2}$ takes constant time. Moreover, once an edge becomes cut, it will not be considered for an edge cut again. So, there can be at most $|P_u|$ edge cuts, which gives a running time of $O(|P_u|)$. ■

Corollary 1: Total cutting cost of curving phase is $C^* \log n$ and the running time is $O(n)$.

Proof: Over all six obtuse triangles $\sum L_u$ is no more than the perimeter of P, which is bounded by C^*, and $\sum |P_u| = n$. Therefore, for all six triangles, the total cost is at most $\sum L_u \log k = C^* \log n$ and running time is $O(\sum |P_u|) = O(n)$. ■

Combining the results of all four phases, we get the following theorem.

Theorem 1: Given a circle Q and a cornered convex polygon P within Q, P can be cut out of Q by using line cuts in $O(n)$ time with a cutting cost of $O(\log n)$ times the optimal cutting cost.

IV. ALGORITHM 2

In this section we present our second algorithm which cuts P out of Q with a constant approximation ratio of 6.48 and running time of $O(n^3)$. This algorithm has three phases: (1) D-separation, (2) cutting out a minimum area rectangle that bounds P and (3) cutting P out of that rectangle by only edge cuts.

The D-separation phase is same as that of our first algorithm and we assume that this phase has been applied.

A. Cutting a minimum area bounding rectangle

We will use the technique of rotating calipers, which is a well known concept in computational geometry and was first introduced by Toussaint [11]. We use the method described by Toussaint [11].

A pair *rotating calipers* remain in parallel. It rotate along the boundary of an object with two calipers being tangents to the object. If the object is a convex polygon P, then one fixed caliper, which we call the *base caliper*, is tangent along an edge e of P and the other caliper is tangent to a vertex or an edge of P. In the next step of the rotation, the base caliper moves to the next edge adjacent to e and continue. The rotation is *complete* when the base caliper has encountered all edges of P.

For our case we use two pairs of rotating calipers, where one pair is orthogonal to the other. We fixed only one caliper, among the four, as the base caliper. As we rotate along the boundary of P, we always place the base caliper along an edge of P and adjust other three calipers as the tangents of P. The four calipers give us a bounding rectangle of P. After the rotation is complete, we identify the minimum area rectangle among the n bounding rectangles. For that rectangle we apply one cut along each of its edges that are not collinear with the chord of the D.

The above technique can be done in $O(n)$ time [11]. Once the base calipers is placed along an edge, the other three calipers are also rotated and adjusted to make them tangent to P. Notice that each caliper "traverses" an edge or a vertex exactly once.

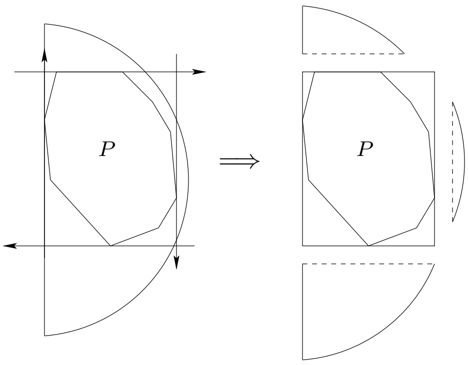


Figure 8. Rotating two pairs of orthogonal calipers. The broken lines are the total cutting cost of this phase.

Lemma 8: The cost of cutting a minimum area rectangle out of the D achieved from the D-separation phase is no more than $2.57C^*$.

Proof: There can be at most four pieces, other than the one inside the bounding rectangle, resulting from four cuts applied to the D. The length of each cut is no more than the portion of the perimeter of D that is separated by that cut. So, as a whole the total cutting cost is no more than the perimeter of D. Also see Fig. 8

Now the perimeter of D is $C_D + R\theta$, where C_D is the length of the chord of D and θ is the angle made by the arc of D at the center c. Since C_D is the cost of D-separation, which by Lemma 4 is bounded by C^*, θ is at most π, and C^* can not be more than $2R$, the maximum perimeter of D is $C^* + (C^*/2)\pi = 2.57C^*$. ■

B. Cutting P out of a minimum area rectangle by only edge cuts

For this phase we simply apply the constant factor approximatoin algorithm of Tan [4]. If P is bounded by a minimum area rectangle, then Tan's algorithm cuts P out of the rectangle by using only edge cuts in $O(n^3)$ time and with approximation ratio $(1.5 + \sqrt{2})$ [4].

We summerize the result of our second algorithm in the following theorem.

Theorem 2: Given a circle Q and a cornered convex polygon P of n edges within Q, P can be cut out of Q by using line cuts in $O(n^3)$ time with a cutting cost of 6.48 times the optimal cutting cost.

Proof: We have the cutting cost of C^* for D-Separation, $2.57C^*$ for cutting the minimum area rectangle, and $(1.5+\sqrt{2})C^*$ for cutting P out of the rectangle, which give a total cost of $6.48C^*$. ■

V. CONCLUSION

In this paper we have given two algorithms for cutting a convex polygon P out of a circle Q by using line cuts where P resides in one side of a diameter of Q. Our first algorithm is an $O(\log n)$-approximation algorithm with $O(n)$ running time, where n is the number of edges of P. Our second algorithm is a 6.48-approximation algorithm with running time $O(n^3)$.

While there exist several algorithms when Q is another polygon, we are the first to address the problem where Q is a circle. Our first algorithm has a better running time and the second one has better approximation ratio than the best known previous algorithms that deal with Q as a convex polygon.

There remain several open problems and directions for future research:

1) The general case of this problem where P is not necessarily in one side of a diameter of Q is still to be solved.
2) We also think it would be interesting to see approximation schemes for this problem.
3) Finally, in many industry applications it is common to have both P and Q as 3D objects, for which we do not know any algorithm. In future it is important to design similar algorithms for the 3D case.

REFERENCES

[1] J. Bhadury and R. Chandrasekaran, "Stock cutting to minimize cutting length," *European Journal of Operations Research*, vol. 88, pp. 69–87, 1996.

[2] E. D. Demaine, M. L. Demaine, and C. S. Kaplan, "Polygons cuttable by a circular saw," *Computational Geometry: Theory and Applications*, vol. 20, pp. 69–84, 2001.

[3] M. H. Overmars and E. Welzl, "The complexity of cutting paper," in *Proc. 1st Annual ACM Symposium on Computational Geometry (SoCG'85)*, 1985, pp. 316–321.

[4] X. Tan, "Approximation algorithms for cutting out polygons with lines and rays," in *Proc. 11th International Conference on Computing and Combinatorics (COCOON'05)*, ser. LNCS, vol. 3595. Springer, 2005, pp. 534–543.

[5] A. Dumitrescu, "The cost of cutting out convex n-gons," *Discrete Applied Mathematics*, vol. 143, pp. 353–358, 2004.

[6] ——, "An approximation algorithm for cutting out convex polygons," *Computational Geometry: Theory and Applications*, vol. 29, pp. 223–231, 2004.

[7] O. Daescu and J. Luo, "Cutting out polygons with lines and rays," *International Journal of Computational Geometry and Applications*, vol. 16, pp. 227–248, 2006.

[8] S. Bereg, O. Daescu, and M. Jiang, "A PTAS for cutting out polygons with lines," in *Proceedings of the 12th Annual International Conference on Computing and Combinatorics (COCOON'06)*, ser. LNCS, vol. 4112. Springer, 2006, pp. 176–185.

[9] R. Chandrasekaran, O. Daescu, and J. Luo, "Cutting out polygons," in *Proc. 17th Canadian Conference on Computational Geometry (CCCG'05)*, 2005, pp. 183–186.

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, Massachusetts: MIT Press, 2001.

[11] G. Toussaint, "Solving geometric problems with the rotating calipers," in *Proc. 2nd IEEE Mediterranean Electrotechnical Conference (MELECON'83)*, Athens, Greece, 1983.

Syed Ishtiaque Ahmed was born in 1986 in Pabna, Bangladesh. He completed his B.Sc. Engineering in Computer Science and Engineering from Bangladesh University of Engineering and Technology (BUET) in March, 2009. Currently he is an M.Sc. student in the same department. He is also working as a Lecturer in United International University, Dhaka, Bangladesh. During his undergraduate studies, he got University Merit Scholarship, Deans List Award, Imdad Siatara Khan Foundation Scholarship etc. for outstanding academic performances. His research interests have focused on the problems of Computational Geometry.

Md. Ariful Islam was born in Bogra, Bangladesh in 1987. He received his B.Sc. Engineering in Computer Science and Engineering from Bangladesh University of Engineering and Technology (BUET) in March, 2009. While studying as an undergrad student, he got University Merit Scholarship, Dean's List Award, Educational Board Scholarship. He was the moderator of BuetNet (Association of BUET internal network). He is currently working as a Quantitative Software Developer at Stochastic Logic Limited, Bangladesh. At the same time, he is doing his M.Sc. in computer science and engineering in BUET. His research interests include Computational Geometry, Data Mining, and Artificial Intelligence.

Masud Hasan was born in Kurigram, Bangladesh in 1973. He received his B.Sc. and M.Sc. Engineering in Computer Science and Engineering from Bangladesh University of Engineering and Technology (BUET) in 1998 and 2001, respectively. He received his PhD in Computer Science from the University of Waterloo, Canada in 2005. In 1998, he joined the Department of Computer Science and Engineering of BUET as a Lecturer, and since 2001 he is an Assistant Professor in the same department. His undergraduate and graduate teaching includes Algorithms, Graph Theory, Mathematical Analysis of Computer Science, Computational Geometry, and Algorithms in Bioinformatics. His main research interests focus on Algorithms, Computational Geometry, and Algorithms in Bioinformatics. He is the author of more than 25 peer-reviewed international journal and conference papers. He has served as a reviewer for many international journal and conferences.