A Fast Approximation Algorithm for TSP with Neighborhoods and Red-Blue Separation

Joachim Gudmundsson Christos Levcopoulos

Department of Computer Science
Lund University, Box 118, S-221 00 Lund, Sweden

Abstract. In TSP with neighborhoods (TSPN) we are given a collection X of k polygonal regions, called *neighborhoods*, with totally n vertices, and we seek the shortest tour that visits each neighborhood. The Euclidean TSP is a special case of the TSPN problem, so TSPN is also NP-hard. In this paper we present a simple and fast algorithm that, given a start point, computes a TSPN tour of length $O(\log k)$ times the optimum in time $O(n+k\log k)$. When no start point is given we show how to compute a "good" start point in time $O(n^2 \log n)$, hence we obtain a logarithmic approximation algorithm that runs in time $O(n^2 \log n)$. We also present an algorithm which performs at least one of the following two tasks (which of these tasks is performed depends on the given input): (1) It outputs in time $O(n \log n)$ a TSPN tour of length $O(\log k)$ times the optimum. (2) It outputs a TSPN tour of length less than $(1+\epsilon)$ times the optimum in cubic time, where ϵ is an arbitrary real constant given as an optional parameter.
The results above are significant improvements, since the best previously known logarithmic approximation algorithm runs in $\Omega(n^5)$ time in the worst case.

1 Introduction

A salesman wants to meet potential buyers. Each buyer specifies a region, his *neighborhood*, within which he is willing to meet. The salesman wants to find a tour of shortest length that visits all of the buyers neighborhoods and finally returns to his initial departure point. The TSP with neighborhoods (TSPN) asks for the shortest tour that visits each of the neighborhoods. The problem generalizes the Euclidean Traveling Salesman Problem in which the areas specified by the buyers are single points, and consequently it is NP-hard [6, 12] (for approximability results concerning TSP see [3]).

One can think of the TSPN as an instance of the "One-of-a-Set TSP" also known as the "Multiple Choice TSP", and the "Covering Salesman Problem". This problem is widely studied for its importance in several applications, particularly in communication network design and VLSI routing [9, 14]. Arkin and Hassin [1] gave an $O(1)$-approximation algorithm for the special case in which the neighborhoods all have diameter segments that are parallel to a common direction, and the ratio between the longest and the shortest diameter is bounded by a constant. Recently, Mata and Mitchell [11] provided a general framework

that gives an $O(\log k)$-approximation algorithm for the general case, when no start point is given, with polynomial time complexity $\Omega(n^5)$ in the worst case. In this paper we give several results: First we show a simple and practical algorithm that produces a TSPN tour with logarithmic approximation factor in the case when a start point is given. If no start point is given we show how a good start point can be computed in $O(n^2 \log n)$. Hence, by combining these two results we obtain a logarithmic approximation algorithm for the general case with running time $O(n^2 \log n)$. Our main result is an algorithm that, given an arbitrary real constant ϵ as an optional parameter, performs at least one of the following two tasks (depending on the instance): (1) It outputs in time $O(n \log n)$ a TSPN tour of length $O(\log k)$ times the optimum. (2) It outputs a TSPN tour of length less than $(1+\epsilon)$ times the optimum in cubic time.

The first part of our method builds upon the idea in [11], in that our logarithmic approximation algorithm produces a guillotine subdivision. However, we produce a quite different guillotine partition (partly inspired from [7, 10]) and show that it has some nice "sparseness" properties, which guarantee the $O(\log k)$ approximation bound. The method described can also be applied to other problems as suggested by [11]. We also consider the Red-Blue Separation Problem (RBSP). The RBSP asks for the minimum-length simple polygon that separates a set of red points from a set of blue points, a problem shown to be NP-hard [2, 5]. Mata and Mitchell showed that the general framework presented in [11] gives an $O(\log m)$-approximation algorithm for this problem in time $O(n^5)$, where $m<n$ is the minimum number of sides of a minimum-perimeter rectilinear polygonal separator for the points. By using the methods suggested in this paper we obtain an $O(\log m)$ approximation algorithm that runs in time $O(n \log n)$.

This paper is organized as follows. In Section 3 the approximation algorithm is presented for the TSPN case where a start point is given. First we compute a bounding square that includes or touches all neighborhoods. Next, a guillotine subdivision algorithm operating within this square is presented that runs in time $O(n+k \log k)$. We prove that the subdivision is of length $O(\log k)$ times the length of an optimal tour. In Section 4 we show how to compute start points that are included in TSPN-tours of length within a constant times the optimal in time $O(n^2 \log n)$. In Section 5, we describe an algorithm that, depending on the given input, decides which method to apply to obtain a TSPN tour. Finally, in Section 6, we show how the methods presented in Section 3-4 can be used to obtain approximation algorithms for some separation problems. The algorithms presented in Section 3-4 and 6 are very practical and easy to implement. The TSPN algorithm has been implemented and runs efficiently even for quite large instances [8]. The proofs omitted in this extended abstract can be found in [8].

2 Definitions and Preliminaries

Throughout this paper we will denote by X the collection of k possibly overlapping simple polygons, called *neighborhoods*, with totally n vertices in the plane.

A polygonal subdivision D of a polygon P is said to be "guillotine" if it is a binary planar partition of P, i.e. either there exist no edges of D interior to P, or there exists a straight edge of D such that this edge is a chord of P dividing P into two regions, P_1 and P_2, such that D restricted to P_i, $1 \leq i \leq 2$, is a guillotine subdivision of P_i. If all the faces of the subdivision are rectangles, we obtain a guillotine rectangular subdivision. From now on we will denote by GRS the guillotine rectangular subdivision obtained from the procedure described in Section 3.1. We will denote by $|p|$ the total length of the segments of p, where p may be a polygon, a tour or a subdivision.

3 A Fast Approximation Algorithm for TSPN

This section is devoted to the proof of the following theorem.

Theorem 1. *Let X be a collection of k possibly overlapping simple polygons, having a total of n vertices. Let SP be any start point. One can compute in time $O(n+k \log k)$ a TSPN tour starting at SP, whose length is $O(\log k)$ times the length of a minimum-length tour starting at SP.*

The general idea is simple. Compute a bounding square that includes or touches all neighborhoods. Next a binary partition is performed within the square such that every neighborhood is intersected either by the binary partition or the bounding square. By traversing the bounding square and the binary partition we obtain a tour. We show how to do this in some more details.

First we detect each neighborhood whose closed region contains SP. This is done in linear time by using standard techniques (see, e.g. [13]). These neighborhoods are visited without departing from SP, and we may exclude them from further consideration. The next step is to compute the minimal isothetic square, called *bounding square*, or $\mathcal{B}$, with center at SP such that each neighborhood is at least partially contained in or touched by $\mathcal{B}$. This bounding square can be constructed in linear time by computing, for each neighborhood $x_i \in X$, $1 \leq i \leq k$, the minimum isothetic square centered at SP and reaching x_i, and then taking the maximum over all these squares, Fig. 1a.

If all the polygons intersect the bounding square then we output the tour obtained by walking from SP to the bounding square, then following the perimeter of the bounding square and returning to SP. This tour is at most five times longer than an optimal TSPN-tour. (In the optimal tour, the salesman has to reach the boundary of $\mathcal{B}$ and then return.) Otherwise, one or more neighborhoods lie entirely inside $\mathcal{B}$. In this case we construct a rectangular binary planar partition, which we call GRS, of the bounding square $\mathcal{B}$, such that every polygon lying entirely within $\mathcal{B}$ is intersected or touched by the GRS. In [11], Mata and Mitchell described how a guillotine subdivision and its bounding box can be traversed such that the obtained tour visits all neighborhoods in X and has length at most twice the length of the subdivision plus $|\mathcal{B}|$.

To prove Theorem 1 it remains to define the GRS, show how it can be computed in time $O(n+k \log k)$ (section 3.1), and prove that its length is $O(\log k)$ longer than the length of a shortest tour (section 3.2).

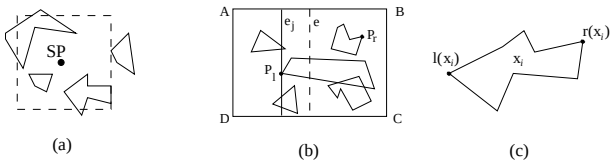

Fig. 1. (a) Constructing a bounding square. (b) A guillotine cut e_j. (c) $l(x_i)$ and $r(x_i)$ of a polygon x_i.

3.1 The guillotine rectangular subdivision

In this subsection we first describe how the subdivision algorithm works. Then, we show how this algorithm can be implemented to run in time $O(n+k\log k)$.

The partition is done as follows. Let T_j be the rectangle in the GRS induced by all segments drawn by the procedure up to but not including the segment e_j, such that e_j is drawn in T_j during the next step. Let A, B, C and D be the corners of T_j in clockwise order, such that AB is the top side of T_j. The segment e_j is drawn parallel to a shortest side of T_j, and it is vertical if T_j is a square. Assume for simplicity that $|AD|\leq|AB|$, i.e. e_j will be a vertical segment, Fig. 1a.

Let e be the vertical chord splitting T_j into two rectangles of equal size. Let $N=\{x_1,\ldots,x_{k'}\}$ be the set of k' polygonal neighborhoods entirely inside T_j, and let $l(x_i)$ respectively $r(x_i)$, $1\leq i\leq k'$, be the leftmost, respectively rightmost, point in the polygonal neighborhood x_i, Fig. 1b. Two cases may occur:

1. If the region to the left of e is empty then the procedure draws e_j such that it intersects the leftmost point $r(x_i)$, $1\leq i\leq k'$. The corresponding procedure is performed symmetrically if the region in T_j to the right of e is empty.

2. If none of the regions are empty then let N_l, respectively N_r, be the polygons in N with non-empty intersection with the closed region of T_j to the left, respectively to the right, of e. Let P_l be the point in $\{\forall x\in N_l \quad l(x)\}$ with shortest distance to e (ties are broken arbitrarily). Symmetrically, P_r is the point in $\{\forall x\in N_r \quad r(x)\}$ with shortest distance to e. If the horizontal distance between P_l and e is shorter than the horizontal distance between P_r and e, then the procedure draws the vertical chord e_j such that it passes through P_l. Otherwise, it draws the vertical chord e_j such that it passes through P_r.

This is done recursively within the two resulting rectangles until all the polygonal neighborhoods are intersected. The final result is called the GRS.

Lemma 1. *The guillotine algorithm presented above can be implemented to run in time $O(n+k\log k)$.*

In [8] we give a detailed description of the algorithm and prove Lemma 1.

3.2 Proving a logarithmic bound

Consider a minimum-length tour, L, with start point SP, and let B_L be its minimal isothetic bounding box. Let $\mathcal{B}$ be the bounding square of X computed

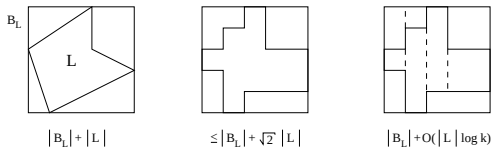

Fig. 2.

by the algorithm, as described above, and let L' be a shortest tour in $\mathcal{B}$. Since the length of the segments of a guillotine rectangular subdivision can be $\Omega(\sqrt{n})$ times the length of the perimeter of the bounding box, we first have to prove that there exists a tour in $\mathcal{B}$ of length $O(|L|)$. Then we show that our subdivision is within a logarithmic factor longer than a shortest tour in $\mathcal{B}$.

Proposition 1. *If $\mathcal{B}$ is a bounding square with perimeter of length within a constant factor longer than the perimeter of B_L, and $\mathcal{B}$ includes a point in an optimal tour, then there exists a TSPN-tour of length $O(|L|)$ within $\mathcal{B}$.*

Recall that the perimeter of the bounding square produced by the algorithm is within a constant factor (4) longer than the perimeter of a minimal bounding box. Note that a minimum-length tour is a simple (not self-intersecting) polygon, having at most k vertices. This result also holds for L'. It is easy to see that a shortest rectilinear TSPN-tour L_R, including the start point SP, is a simple rectilinear polygon and has length at most $\sqrt{2}|L|$. Furthermore, in [4, 10, 11], it was shown that there exists a guillotine rectangular subdivision of the rectilinear tour L_R and its minimal bounding box B_{L_R} such that the length of the subdivision is $O(|L_R| \log k)$, Fig. 2. Hence, we obtain the following fact:

Fact 1. *If $OPT(X, B_L)$ is a rectangular subdivision of minimum length of X and B_L, then it holds that $|OPT(X, B_L)|$=$O(|L| \log k)$.*

According to the above results we have that Fact 1 also holds if we replace B_L and L with $\mathcal{B}$ and L'. This means that an optimal rectangular subdivision of the polygonal neighborhoods X within the bounding square $\mathcal{B}$ is within a logarithmic factor longer than an optimal TSPN tour. Hence, to prove Theorem 1 it remains to show that the GRS presented in the previous section produces a subdivision which is at most within a constant factor longer than an optimal subdivision.

Lemma 2. *Let L' be a minimum-length tour within a bounding square $\mathcal{B}$, let L be an optimal tour and let B_L be its minimal isothetic bounding box. If the length of the perimeter of $\mathcal{B}$ is within a constant factor longer than the perimeter of B_L, then it holds that the guillotine rectangular subdivision of X within $\mathcal{B}$ is of length $O(|L'| \log k)$.*

4 Finding a Good Start Point

In the previous section we showed a fast algorithm that produces a TSPN tour when a start point is given. To use this algorithm in the general case we will in

this section show how to compute a nearly optimal start point, i.e. a start point that is included in a TSPN-tour of length $O(|L|)$, where L is a minimum-length tour of X. We describe a method which, given an arbitrary straight-line segment, computes in time $O(n \log n)$ a starting point on this segment which is a nearly optimal point under the condition that the optimal tour has to pass through this segment. By observing that there is always an optimal tour with non-empty intersection with the boundary of at least one neighborhood, we obtain as a corollary an $O(n^2 \log n)$-time method for finding a globally nearly-optimal start point. Combining this result with the algorithm described in the previous section we obtain a logarithmic approximation algorithm that runs in time $O(n^2 \log n)$.

4.1 The length function

The idea is that we will for each of the n edges of the neighborhoods find the minimum L_∞-distance to all polygonal neighborhoods. This is done by constructing a length function $L(g, f)$ for each pair of edges $g, f \in E$ (not belonging to the same neighborhood), where E is the set of edges in X. This function describes the shortest distance in L_∞-metric from each point on g to f. Let $g, f \in E$ be two edges in E. We want to calculate $L(g, f)$, that is for each point in g we want to know the shortest distance to f in L_∞-metric. Rotate g and f such that g is horizontal. The endpoints of g are denoted G_1 and G_2. The length function is a piecewise linear function containing at most three different pieces. We will calculate the value of $L(g, f)$ in at most four points. Compute the values for G_1 and G_2. Next compute the shortest distance from f's endpoints to g. Denote the points on f with shortest distance to G_1 respectively G_2, by p_1 respectively p_2. $L(g, f)$ is obtained by drawing straight line segments between these four points, from left to right. Note that p_1 and/or p_2 may coincide with the endpoints of g. This function may now be used to compute the shortest distance between an edge $f \in E$ and a neighborhood $x \in X$, where $f \notin x$. First compute the length function for each edge in x and the edge f. Since we only are interested in the shortest distance between every point on f and the neighborhood x, we calculate the lower envelope of these $|x|$ functions, denoted $LEnv(f, x)$. Calculating the lower envelope of the $|x|$ functions can be done in time $O(|x| \log |x|)$ according to Sharir in [15]. If f intersects the closed region described by x then we have to adjust the lower envelope. This can easily be done in linear time.

This new piecewise linear function describes the shortest distance between every point on f and the polygon x, and can be computed in time $O(|x| \log |x|)$.

4.2 Using the length function to obtain a good start point

We extend the above computation such that we obtain a function that for each point on f describes the shortest distance to all the neighborhoods in X.

The algorithm is just an extension of the above described algorithm. Let f be an edge as above and let $\{x_1, \ldots, x_k\}$ be the neighborhoods in X. For each neighborhood $x_i \in X$ and $f \notin x_i$ compute the lower envelope, $LEnv(f, x_i)$, as described above. Combine the $k-1$ lower envelopes for all the neighborhoods

in X to an upper envelope, $UEnv(f, X)$. This new function describes the shortest distance between each point in f and all the neighborhoods in X. In the case when we know that an optimal TSPN-tour intersects f we can select the point on f with minimum $UEnv(f, X)$-value as start point (SP). Note that the function also gives us the size of an optimal bounding square with center at SP.

Calculating the envelope can be done in time $O(n \log n)$ [15]. The idea is to partition the collection of functions into two subcollections. Then calculate recursively the envelopes of each subcollection separately. Finally merge the two partitions into a single refined partition. The total time complexity for this algorithm is $O(n \log n)$.

If the $UEnv$-function is computed for every edge in E we can in linear time find a good start point for the general case, by just selecting the best of the n suggested start points. We obtain the following results.

Proposition 2. *The computed bounding square centered at SP includes a tour of length $O(|L|)$ and the length of the perimeter of the square is at most four times the length of a minimum-length TSPN-tour.*

Hence the computed bounding square fulfills the requirement needed for Lemma 2 to hold and we obtain the following corollary.

Corollary 1. *Let X be a collection of k possibly overlapping simple polygons, having a total of n vertices. One can compute in time $O(n^2 \log n)$ a start point that is included in a tour of length within a constant factor times the optimal.*

5 TSPN When no Start Point is Given

We have already described an approximation algorithm for the general case. By finding a start point and then applying the GRS we obtain an approximation algorithm that runs in time $O(n^2 \log n)$ and produces a tour that is within a logarithmic factor longer than an optimal tour. In this section we describe an algorithm that, depending on the given input, decides which method to apply to obtain a TSPN tour. The decision depends on the value of ϵ, an optional parameter given as input, and the ratio between the size of the smallest neighborhood in X and the length of a minimum-length tour. Recall that the minimal bounding box, B_L, is the smallest axis-aligned rectangle that contains L, where L is a minimum-length tour of X. The algorithm works as follows.

First a minimal axis-aligned bounding box for every neighborhood in X is constructed. Let $x \in X$ be the neighborhood, such that its bounding box, B_x, has shortest long side. Let c denote the center of B_x, and let l be the length of B_x's longest side, Fig. 3a. Next, a minimal isothetic bounding square $\mathcal{B}$ with center at c is constructed such that each neighborhood is at least partially contained in or touched by $\mathcal{B}$, Fig. 3b. This is done in linear time as described in Section 3. Two main cases may occur.

1. If $|\mathcal{B}| > 4l$ then it holds that there exists a neighborhood $x' \in X$ such that x' lies outside B_x. Hence, an optimal tour must intersect B_x, that means that we can find a good start point, SP, that lies on one of the four sides of B_x. This

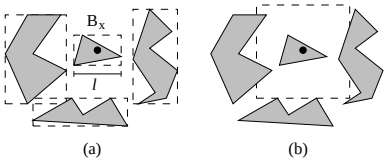

Fig. 3.

can be done in time $O(n \log n)$ as explained in Section 4.2. By applying the subdivision algorithm defined in Section 3.1 with SP as start point we obtain a tour of length $O(\log k)$ times the optimal in time $O(n \log n)$.

2. If $|\mathcal{B}|<4l$ we first make a refined search in time $O(n)$ for much smaller bounding boxes in the vicinity of x. For this reason, we imagine a square-grid centered at c, with side length $2l$, and where each square has side length equal to $l/10$. For each one of the $O(1)$ corners of the small squares, we find the minimal isothetic bounding square, as if this corner was a given starting point. Let D be the smallest of these bounding squares and, let l' be the side length of D. Again, two cases may occur:

(a) If $l'>l/3$, then output D as the TSPN tour. We can show that it is within a constant factor from the minimum, and we have found it in linear time.

(b) If $l'\leq l/3$ then we run the approximation scheme described in Section 5.1. We obtain a TSPN tour of length $1+\epsilon$ in cubic time.

5.1 An approximation scheme

In this section we show how to obtain a polynomial-time approximation scheme, in the case when $l'<l/3$ (defined in the previous paragraph).

Observation 1. *For every TSPN tour of length less than $2l$, it holds that the convex hull of that tour also is a TSPN tour.*

Now, we find (at most) a quadratic number of approximately minimal bounding squares (i.e. with perimeter at most a constant factor larger than the minimal tour), possibly overlapping, such that an optimal tour has to be totally inside at least one of them. This is done as follows. Find all *essential points* in X. These points are the vertices of the polygons of X and the intersections between the polygons. Since there are n edges there can be $O(n^2)$ such intersections. Hence the number of essential points is $O(n^2)$. Then the algorithm selects every minimal bounding square with center at an essential point and with perimeter at most a constant factor longer than a minimal tour (a constant factor longer than the smallest found bounding square). Finally, for reasons explained in the proof of Observation 2, expand each selected minimal bounding square such that the sides of a square is a constant C, say 10, times the length of the original bounding square. There are $O(n^2)$ such bounding squares and each square takes $O(n)$ time to construct. Thus, the time complexity of this step is $O(n^3)$.

Observation 2. *An optimal tour is included in at least one of the $O(n^2)$ bounding squares produced above.*

For each selected bounding square b, let $l(b)$ be the length of the minimum tour overlapping with b and denote by $h(b)$ the length of the sides of b. For each b, we find a tour of length less than $(1+\epsilon)\cdot l(b)$ in linear time, and in this way in cubic total time we find a tour of length $(1+\epsilon)$ times the optimal.

A simple, although impractical, linear-time method to find a tour of length $(1+\epsilon)\cdot l(b)$, is as follows: We partition b into $f(\epsilon)\times f(\epsilon)$ equal-sized squares, where $f(\epsilon)$ is a sufficiently large constant only depending on ϵ. Let S be the set of all points which are corners of these small squares. Now, for each subset S' of S, we construct the convex hull of S'. Next, among those of the constructed convex hulls which are TSPN tours, we select the shortest one. (Of course, in practice we don't really have to consider all subsets of S.) It is easily seen that by choosing $f(\epsilon)$ sufficiently large, say, $f(\epsilon)=\frac{4h}{\epsilon}$, we will find a tour of length $(1+\epsilon)\cdot l(b)$.

6 Other Applications

The methods described in this paper apply also to other problems. Here we show how to use our method for the Red-Blue Separation Problem (RBSP). The objective is to find a minimum-length simple polygon that separates a set of red points, R, from a set of blue points B. This problem is seen to be NP-hard by using a reduction from the Euclidean TSP [2, 5]. In this section we give the idea of an $O(\log m)$ approximation algorithm for RBSP, where $m<n$ is the minimum number of sides of a minimum-perimeter rectilinear polygonal separator for the points. The total number of points is $n=|R|+|B|$. We will solve two problems and pick the shorter length: find a minimum-length polygon that encloses red, while excluding blue; and find a minimum-length polygon that encloses blue, while excluding red.

First construct the minimum bounding box for the red points respectively the blue points. Let P be a minimum-length separating simple polygon, of length $|P|$. Without loss of generality, assume that P surrounds the blue points and excludes the red points. Let $\mathcal{B}$ be the bounding box for B. Note that P lies within $\mathcal{B}$ and that $\mathcal{B}$ is the bounding box of P. If no red points lie within $\mathcal{B}$ then we are finished. The length of the perimeter of this polygon is at most a factor $\sqrt{2}$ longer than $|P|$. Otherwise, one or more red points lie within $\mathcal{B}$. In this case construct a guillotine rectangular subdivision of $\mathcal{B}$, such that every rectangle within the subdivision only contains points of one color. Next we run the GRS-algorithm described in Section 3.1, with some adjustments [8].

In [11], Mata and Mitchell described how a guillotine subdivision and its bounding box can be traversed such that the resulting tour encloses all blue points, excludes all red points, and has length at most twice the length of the subdivision plus $|\mathcal{B}|$. Hence, we obtain the following corollary.

Corollary 2. *Given a set of n points in the plane, where each point is either red or blue, a red-blue separating simple polygon of length $O(\log m)$ times the mini-*

mum can be computed in time $O(n \log n)$, where $m<n$ is the minimum number of sides of a minimum-perimeter rectilinear polygonal separator for the points.

7 Conclusion and Future Work

We have presented approximation algorithms for the TSPN-problem and the red-blue separation problem. For both these results we improved the running-time considerably by computing a simple and fast guillotine rectangular subdivision.

There are several open questions concerning these problems. It is natural to ask whether a constant approximation ratio is achievable for these problems. Further, is there a logarithmic approximation algorithm for TSPN in the case when the neighborhoods are not simple polygons, for example, sets in the plane?

References

1. E. M. Arkin and R. Hassin. Approximation algorithms for the geometric covering salesman problem. *Discrete Applied Math*, 55:197–218, 1994.
2. E. M. Arkin, S. Khuller, and J. S. B. Mitchell. Geometric knapsack problems. *Algoritmica*, 10:399–427, 1993.
3. S. Arora. Polynomial time approximation schemes for Euclidean TSP and other geometric problems. In *FOCS'96*, pages 2–11, 1996.
4. D. Z. Du, L. Q. Pan, and M. T. Shing. Minimum edge length guillotine rectangular partition. Technical Report 02418-86, Math. Sci. Res. Inst., University of California, Berkeley, CA, 1986.
5. P. Eades and D. Rappaport. The complexity of computing minimum separating polygons. *Pattern Recognition Letters*, 14:715–718, 1993.
6. M. R. Garey, R. L. Graham, and D. S. Johnson. Some NP-complete geometric problems. In *STOC'76*, pages 10–21, 1976.
7. T. Gonzalez and S. Q. Zheng. Bound for partitioning rectilinear polygons. In *SoCG'85*, pages 281–287, 1985.
8. J. Gudmundsson and C. Levcopoulos. A fast approximation algorithm for TSP with neighborhoods and red-blue separation. Technical Report LU-CS-TR:97-196, Dept. of Computer Science, Lund University, Lund,Sweden, 1997.
9. J. Hershberger and S. Suri. A pedestrian approach to ray shooting: Shoot a ray, take a walk. *Journal of Algorithms*, 18:403–431, 1995.
10. C. Levcopoulos. Fast heuristics for minimum length rectangular partitions of polygons. In *SoCG'86*, pages 100–108, 1986.
11. C. S. Mata and J. S. B. Mitchell. Approximation algorithms for geometric tour and network design problems. In *SoCG'95*, pages 360–369, 1995.
12. C. H. Papadimitriou. Euclidean TSP is NP-complete. *TCS*, 4:237–244, 1977.
13. F.P. Preparata and M.I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, New York, NY, 1985.
14. G. Reich and P. Widmayer. Beyond steiner's problem: A vlsi oriented generalization. In *Proceedings of the 15th International Workshop Graph-Theoretical Concepts in Computer Science*, volume 411 of *LNCS*, pages 196–210, 1989.
15. M. Sharir. Davenport-schinzel sequences and their geometric applications. In R. A. Earnshaw, editor, *Theoretical Foundations of Computer Graphics and CAD*, volume F40 of *NATO ASI*, pages 253–278. Springer-Verlag, Berlin, Germany, 1988.