The Shortest Separating Cycle Problem

Esther M. Arkin[1], Jie Gao[1], Adam Hesterberg[2], Joseph S. B. Mitchell[1], and Jiemin Zeng[1]

[1] Stony Brook University, Stony Brook, NY, USA. esther.arkin, jie.gao, joseph.mitchell. jiemin.zeng@stonybrook.edu
[2] Massachusetts Institute of Technology, Boston, MA, USA. achester@mit.edu

Abstract. Given a set of pairs of points in the plane, the goal of the shortest separating cycle problem is to find a simple tour of minimum length that separates the two points of each pair to different sides. In this article we prove hardness of the problem and provide approximation algorithms under various settings. Assuming the Unique Games Conjecture, the problem cannot be approximated within a factor of 2. We provide a polynomial algorithm when all pairs are unit length apart with horizontal orientation inside a square board of size $2 - \varepsilon$. We provide constant approximation algorithms for unit length horizontal or vertical pairs or constant length pairs on points laying on a grid. For pairs with no restriction we have an $O(\sqrt{n})$-approximation algorithm and an $O(\log n)$-approximation algorithm for the shortest separating planar graph.

Keywords: Shortest separating cycle, traveling salesman problem

1 Introduction

Given a set $P = \{(p_i, q_i) | 1 \leq i \leq n\}$ of pairs of points in the plane, we seek a *shortest separating cycle* T, a tour where for every pair of points, one point is inside the tour and the other is outside. Each pair (p_i, q_i) can be represented by a line segment connecting p_i and q_i. Therefore each segment is cut by the tour an odd number of times. Throughout this paper, we use whichever interpretation is more intuitive.

The motivation for this problem originates from data storage and retrieval in a distributed sensor network [16, 18]. Consider an application in which sensors are installed at parking spots to detect if the spot is empty, while mobile users roaming around in the city are in need of such information. We would need to have a data processing, storage and retrieval scheme to allow allow mobile users anywhere to quickly retrieval data of interest. The solution of always delivering the query to the data source may suffer from a single point of failure and traffic bottleneck. Therefore a natural solution is to adopt

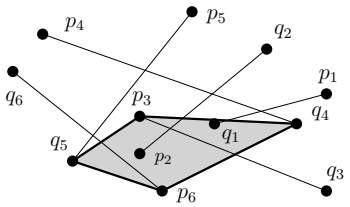

Fig. 1. The shortest separating cycle problem.

geographical hashing. In [18] each data is hashed to two storage sensors by its type. While a piece of data i is delivered from one storage site p_i to the other storage location q_i using multi-hop routing it is convenient for all the nodes on the relay path to also cache the data item. For a mobile user seeking data of a particular type i, the user can issue a query which only needs to visit a node that has cached the data. Say if the user query travels along a tour that separates p_i and q_i, the query will hit the cached data for sure – as any path connecting p_i and q_i is intersected by the tour. In this retrieval scheme one can easily query for a collection of data of multiple types $1, 2, \cdots n$, as long as the query follows a tour that separates each pair of nodes p_i and its corresponding hashed storage node q_i. This becomes precisely the separating cycle problem [18]. Finding the shortest separating cycle is natural, as the shortest tour minimizes energy consumption and delay.

In computational geometry, this problem is related to many traveling salesman problem (TSP) variants, including the red-blue separation problem, TSP with neighborhoods, and one-in-a-set TSP (also known as group TSP), that have been well studied [2, 12]. All of these problems are known to be NP-hard in the Euclidean plane as they all contain the classical TSP as a special case. The shortest separating cycle problem is different from any of these problems. In the red-blue separation problem, given a set of red and blue points in the plane, the aim is to find the shortest tour that separates the blue points from the red points. In our problem, the points in the pairs need to be separated but they are not assigned colors. Thus part of the challenge is to determine which point of each pair is inside the tour and which one is outside.

In the TSP with neighborhoods (TSPN), given a set of regions, the goal is to find the shortest tour that visits each region. One may attempt to connect a line segment for each pair in our input and apply an algorithm for TSP with neighborhoods where the neighborhoods are line segments. However, this does not necessarily give a valid solution since the TSP with neighborhoods solution might reflect on the edge and not enclose an endpoint in its cycle such as in Figure 2(i).

For the one-in-a-set TSP, we are given a collection of sets and the problem asks for the shortest tour that visits at least one element in each set. Any one-in-a-set TSP solution to our input can be easily modified to become a separating cycle. However, a separating cycle does not need to visit every point it is including or excluding so the one-in-a-set TSP solution may be excessively long. An example of this can be seen in Figure 2(ii).

Our Results In this paper we are the first to study the shortest separating cycle problem and provide both hardness and approximation results. In particular, we consider special cases where the orientation of the input pairs, the distance between the input pairs, and the configuration of the domain are restricted. We vary the size of the square board the input points are confined within as well as the range of orientations the input pairs have from strictly horizontal and/or vertical to any orientation. Some cases have additional restrictions such as how

far each pair of points are from each other and whether or not the input points must lie on a grid. The results are summarized in Tables 1.

In general, despite the apparent similarity and connection to many other TSP variants which have easy constant-factor approximation results, the shortest separating cycle problem is a lot harder. Many ideas that were used in typical TSP algorithms are not applicable here. Indeed, Indeed, we show that the problem is hard to approximate for a factor of 1.3606 unless P=NP and is hard to approximate better than a factor of 2 assuming the Unique Games Conjecture. We provide a polynomial time algorithm when all pairs are unit length horizontal segments inside a square board of size $2-\varepsilon$. We provide approximation algorithms for unit length horizontal or vertical segments or constant length segments on points laying on a grid. These scenarios are of particular interest to the application setting in a sensor network. Last, for arbitrary pairs we have an $O(\sqrt{n})$-approximation algorithm and an $O(\log n)$-approximation algorithm for the shortest separating planar graph problem, in which the objective is to compute an embedded planar graph of minimum total edge length so that the two endpoints of each pair are in different faces.

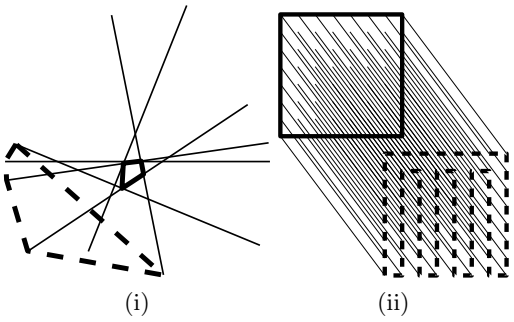

Fig. 2. Shortest Separating Cycle is different from TSPN or one-in-a-set TSP. (i) The TSP with neighborhoods solution (solid) is not a valid separating cycle and is a much shorter tour than the shortest separating cycle (dashed). (ii) The one-in-a-set TSP solution (dashed) is a much longer tour than the shortest separating cycle (solid).

Board Size	Unit Length Horizontal	Unit Length Horizontal &Vertical	Unit Length Arbitrary Orient.	Constant Length Arbitrary Orient. Points on Grid
$2-\varepsilon$	in P	4-approx	NP-hard	NA
$M = O(1)$	$O(1)$-approx	$(M^2+1)/4$-approx	Hard to approx	NA
n	$O(1)$-approx	$O(1)$-approx	Hard to approx	$O(1)$-approx

Table 1. Approximation algorithm and hardness results for different settings.

Related Work

TSP. The traveling salesman problem is one of the most well known geometric problems in history. It is one of the first problems known to be NP-hard [7, 15]. In a metric setting Christofides provided a 3/2 approximation algorithm [5]. In the Euclidean setting, the problem is known to admit a PTAS, independently shown by Arora [2] and Mitchell [12].

TSP with Red Blue Separation. The red blue separation problem in the plane admits a PTAS (by [12] or by [3]).

TSP with Neighborhoods. TSP with neighborhoods was first studied by Arkin and Hassin [1] in which $O(1)$-approximation algorithms were developed when the neighborhoods are translates of a convex polygon or when the neighborhoods are unit disks. For general (nondisjoint) connected neighborhoods, an $O(\log n)$-approximation algorithm is known where n is the number of neighborhood regions [11, 8], and it is NP-hard to approximate within a $2 - \varepsilon$ ratio [17]. For fat regions of bounded depth, there is a PTAS [13] (even in doubling metrics [4]), while for general connected regions of bounded depth, or for convex regions, an $O(1)$-approximation is known in two dimensions [14].

One-of-a-set TSP or Group TSP. The one-of-a-set or group TSP is the TSPN in which the neighborhoods are discrete sets of points (and thus disconnected). Safra and Schwartz [17] show the 2D problem is NP-hard to approximate to within any constant factor; for groups that are sets of k points, they also give approximation lower bounds ($\Omega(\sqrt{k})$). Slavík [19] gives a $(3/2)k$-approximation, based on linear programming methods.

2 Hardness

The shortest separating cycle problem is NP-hard by a trivial reduction from the traveling salesman problem (TSP). For any TSP instance with cities at location w_i, we place a pair of points p_i, q_i very close to each w_i. In order to separate the points, the tour will need to visit each city w_i. Thus the shortest separating cycle problem is as hard as TSP. In the following we show stronger results that the problem is hard to approximate.

2.1 Inapproximability for Any Length Segments

Theorem 1. *The shortest separating cycle problem with no restrictions on the distance and orientation of input pairs is NP-hard to approximate better than a factor of 1.3606. It is hard to approximate better than a factor of 2 assuming the unique games conjecture.*

Proof. Our reduction is from minimum vertex cover. Given a graph $G = (V, E)$, the goal of the minimum vertex cover problem is to find a minimum cardinality subset of vertices $V^* \subseteq V$ such that every edge in E is incident to at least one vertex in V^*.

Now we will create a set of pairs in the plane. First we place each vertex in V along a circle with a center w' and designate another location w'' at a distance $\Omega(nm)$ from w'. Here, $n = |V|$ and m is a constant. For every vertex $v \in V$, we place m endpoints overlapping a single point at w' and their corresponding endpoints in a $\sqrt{m} \times \sqrt{m}$ grid pattern around v (as shown by dark dots in Figure 3). Finally, w' and w'' are connected by a long segment. The result is a "wheel" composed of vertices and edges in G with "spokes" towards the center, a hub at w', and a large arm from w' to w''.

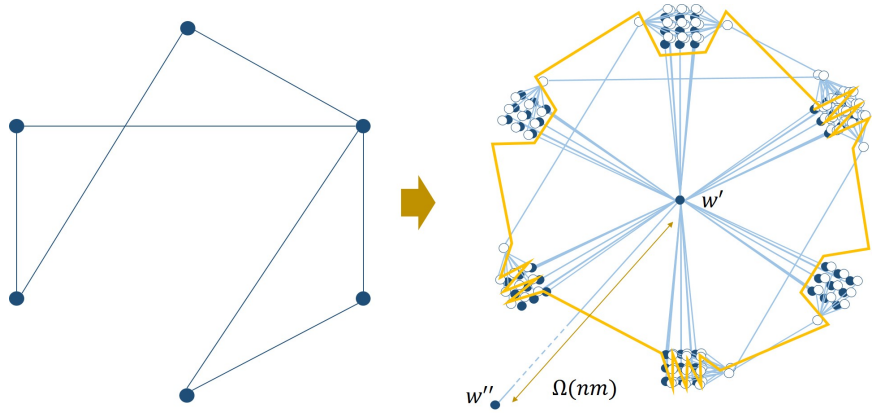

Fig. 3. Reduction from vertex cover.

For every edge $e(u, v) \in E$, we place m endpoints in a $\sqrt{m} \times \sqrt{m}$ grid pattern in a $\sqrt{m} - 1$ square around u and their corresponding endpoints overlapping a single point u_e near the grid. Another grid of m endpoints is placed in a similar grid around v with their corresponding points overlapping on a point v_e. These points are shown in hallow dots in Figure 3. These hallow grid points are extremely close to the dark grid points. Finally, u_e and v_e are connected by a segment. Therefore, for every gadget that represents an edge, exactly one set of grid points must be inside the cycle.

Let's first consider the super long arm from w' to w'', and the pairs created by edges of G. First the separating cycle will need to include some points from the edge gadgets. If the separating cycle also visits w'' (to include it inside), there is an additional cost of length of $O(nm)$, which is so prohibitive and must be avoided. So the separating cycle will include the points at w' in the interior to separate w' and w''. This implies that the dark dots connected to w' by spokes need to be outside the cycle. Now let's consider a set of hollow grid points near a set of dark grid points. As the dark grid points need to be outside, then the hallow points need to be visited invidually to include them inside the cycle. Since points in a grid are unit distance away from each other, a path that visit a grid of points has a length of at least m.

The optimal solution visits the minimum number of vertices in V while also separating all pairs of points which is a minimum vertex cover of G. The parity of the segment chains ensure that when one point of a grid is collected, all points in that grid must be collected.

The length of an optimal separating cycle in this instance is $O(|OPT_{VC}|m + n\sqrt{m})$ where $|OPT_{VC}|$ is the size of the optimal solution of the corresponding vertex cover problem. If $m \geq n$, then the cost of navigating between grids at the vertices of V dominates the cost of the optimal solution. The rest of the cost, $O(n\sqrt{m})$ is from traveling between vertices and collecting w'.

Now we assume that we are given a δ-approximation to our construction of the shortest separator problem. This path has distance at most $O(\delta|OPT_{VC}|m + \delta n\sqrt{m})$. To convert this solution into a solution for the corresponding vertex

cover problem, any grid points collected at a vertex translates to a vertex selected in the vertex cover. This means that any additional vertex above the optimal solution of the vertex cover problem translates to an additional $O(m)$ length in the given approximation of the shortest separator instance. If we let $m = \Omega(\delta^2 n^2)$, then the approximation solution visits at most $\delta|OPT_{VC}|$ vertices and is a δ-approximation for vertex cover. Therefore, the lower bounds for the vertex cover problem apply to the shortest separator problem. This problem cannot be approximated with a factor better than 1.3606 unless $P = NP$ or 2 assuming the unique games conjecture.

3 Algorithms

We describe exact and approximation algorithms for the shortest separating cycle problem under different scenarios.

3.1 Board Size $2 - \varepsilon$, Horizontal and Vertical Unit Segments

In this scenario, all n input segments are inside a square board of size $2 - \varepsilon$ for some $\varepsilon > 0$ and are restricted to have horizontal or vertical orientations. Without loss of generality we assume all the endpoints of different input segments do not share a common x-coordinate or y-coordinate. This can be done by perturbing the input slightly.

First, all unit length boxes which contain *at least* one endpoint of each segment are found. This can be executed in polynomial time by checking all possible combinatorial configurations of unit length boxes. The total number of combinatorial types of such squares is $O(n^2)$ since we can assume without loss of generality that the square always has two input endpoints (from two different segments) on its boundary. Aside from the two points on the boundary, each such box actually contains *exactly* one endpoint of each input segment. For the two boudary points we enumerate all combinations of including these boundary points inside the box. The convex hull of all endpoints inside the box is a candidate separating cycle. We can enumerate all such boxes to find the shortest separating cycle.

If a unit length box that strictly contains one endpoint of every segment cannot be found, then the board is divided into four squares each of size $1 - \varepsilon/2$ and are colored in a checker board pattern. The square are named S_1, S_2, S_3, S_4 in a counter clockwise manner. Consider two squares along the diagonal (named S_1 and S_3, see Figure 4(ii)). We create a tour that walks along the perimeter of their union. To accommodate corner cases, we consider the top and left border of S_3 to be open edges. This generates a curve T' of length $8 - 4\varepsilon$.

Theorem 2. *For the shortest separating cycle problem with a square $2 - \varepsilon$ domain where input pairs are exactly one apart and have either horizontal or vertical orientation, our algorithm outputs a cycle that is a 4-approximation to the optimal solution.*

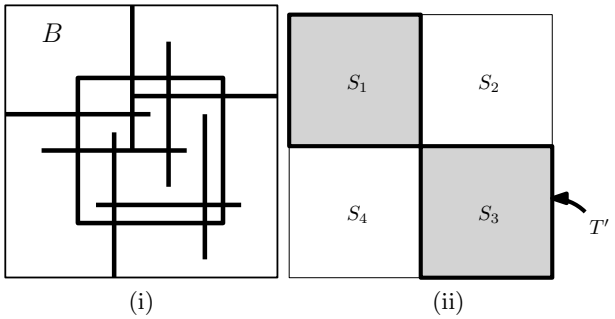

Fig. 4. The two cases in our algorithm for the shortest separating cycle on a $2-\varepsilon$ board. Case (i): Exactly one endpoint of each input pair can be enclosed in a unit square. Case (ii): The curve T' traverses around S_1 and S_3.

Proof. There are two cases in the algorithm which outputs two different types of cycles. In case (i), the optimal solution fits inside a unit length box B. Every segment has one endpoint inside B and B will be discovered in the first phase of the algorithm. The convex hull of the points inside B is the shortest separating tour that contains all points in B.

For case (ii) we know the optimal tour cannot be completely contained by a unit box and therefore must have length at least 2. We first argue that T' is a valid separating cycle. Since each segment has unit length, the two endpoints of each segment cannot fit inside any single square of size $1-\varepsilon/2$ and thus cannot both lie inside $S_1 \cup S_3$ nor inside $S_2 \cup S_4$. Therefore each segment must have exactly one endpoint inside $S_1 \cup S_3$. Therefore, T' is a valid separating cycle. Since T' has length $8-4\varepsilon$ and the optimal tour has length at least 2, T' is a 4-approximation of the optimal solution.

3.2 Constant-Size Boards

We can extend the checkerboard strategy for any constant board size M, where M is an integer. The only modification is that in the second case, a larger checkerboard of $M \times M$ unit squares is used. The squares are colored in a checkerboard manner, and partitioned into white squares and dark squares. To make the tiling a perfect partition of the plane, we consider each square pixel to include its top edge, except for the NE corner, and to include its left edge, except for the SW corner. Again any unit length vertical or horizontal segment has two endpoints in different colored squares. Thus a tour T' that separates the white squares from the dark squares would be a valid separating cycle. Such a tour can always be found by taking a cycle along the boundary of the outermost ring of the dark squares, and iterating towards the center. All the tours can be joined into a single tour of the same length. See Figure 5 for an illustration.

Theorem 3. *We consider the case of the shortest separating cycle problem with an $M \times M$ square domain and where the input pairs are restricted to be exactly one apart with either horizontal or vertical orientation. Our algorithm including the checkerboard strategy is an $(M^2+1)/4$-approximation.*

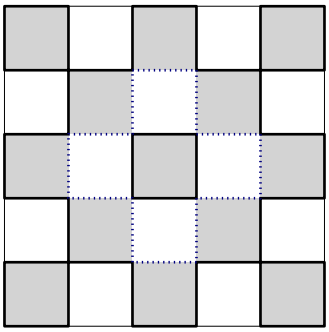

Fig. 5. Constant sized square board with cycles along the perimeter of the dark squares.

Proof. Our proof is similar to the proof of Theorem 2. Either we can find a unit length box containing at least one endpoint of each segment (in which case we find the optimal solution), or the optimal tour has length at least 2. In the second case we will take the tour T' along the perimeter of the union of the dark squares. T' has length at most $M^2/2$ if M is even, and at most $(M^2+1)/2$ if M is odd. Thus the approximation factor is at most $(M^2+1)/4$.

3.3 Any Board Size, Horizontal and Vertical Unit Segments

If the board size M is a constant, we can apply the same checkboard idea as in the previous section. But when M is large, we have to use a different idea to get a constant approximation.

First, we overlay a grid of unit squares over the domain partitioning the domain into light and dark squares in a checkerboard pattern. For each grid cell, we consider the top edge, excluding the NE corner, and the left edge, excluding the SW corner, as closed edges.

We refer to dark squares that have a point (from a pair) in them as "occupied". Let S be the occupied squares. Let S' be the 3-by-3 squares centered on the squares of S. In the following we assume without loss of generality that $|S| \geq 5$. The case $|S| \leq 4$ can have an arbitrarily small optimal value; but this constant-size case can be easily handled.

Now we consider the shortest TSP with Neighborhoods (TSPN) tour on the set S' of enlarged squares and name the length as $TSPN(S')$. This tour connects the regions in S' but we must also separate the pairs of points in each region. We further apply the constant factor approximation to each region in S'. Our algorithm is simply this: Run a TSPN algorithm on S' (for which there is a PTAS [6]), and augment the tour with our approximation algorithm for square regions of constant size.

Theorem 4. *Our algorithm for the shortest separating cycle problem for input pairs restricted to a separation of exactly 1 and only horizontal or vertical orientation, is a constant approximation.*

Proof. We have two cases regarding the size of S, $|S| \geq 5$ and $|S| \leq 4$. In the first case, $|S| \geq 5$, the length of the output is at most $(1+\varepsilon)TSPN(S') + O(|S|)$. Since the optimal solution must visit every enlarged square in S', then $OPT = \Omega(TSPN(S'))$. Assuming no single point stabs all squares of S' (i.e., assuming $|S| \geq 5$), the standard packing argument shows that TSPN on a set of nearly disjoint (i.e., constant depth of overlap), equal-sized squares requires length proportional to the number of squares times the side length of the squares. This leads us to claim that $OPT = \Omega(|S|)$. Therefore, $(1+\varepsilon)TSPN(S')+O(|S|)$ is $O(OPT)$ and our tour is a constant approximation.

For the case where $|S| \leq 4$, our strategy only changes if a single point is contained in all squares of S'. In this case, our entire input can be contained in a 5×5 square and we refer to our algorithm for constant sized domains.

3.4 Any Orientation, Bounded Aspect Ratio

We now assume that the distance between any two points (not restricted to designated pairs) in S are greater than or equal to 1. The segments defined by the pairs of points may have any orientation and their distances are bounded by a constant. Let $r = cL$ for some constant value of $c \geq 1$ where L is the length of the longest segment. The aim is to find a subset I of pairs of points where the shortest distance between any two segments is greater than r and the shortest distance between any input segment and it's closest segment in I is less than r. Then we find the TSPN path on line segments in I. Next, we divide the region into neighborhoods by assigning the remaining segments to their nearest segment in I. The TSPN tour of the segments, $TSPN(I)$, is augmented with detours that separate all of the segments in each neighborhood. The resulting tour is a constant approximation of the optimal separating tour.

To find such an independent set, we randomly select neighborhoods of size $O(r)$ until all segments have been selected. A segment s is randomly chosen and removed from the set of input segments along with all remaining segments within distance (shortest distance) r of s. The segment s is placed in the set I. This procedure is repeated until all segments are removed. The shortest distance between any two segments in I is greater than r. The shortest distance between any segment and it's closest segment in I is less than or equal to r. Each segment is assigned to it's closest segment in I. The set of segments assigned to a segment s in I is denoted as $N(s)$.

A tour is constructed by first finding a TSPN tour on I. Then the path is augmented by a shorter tour within each neighborhood of each segment in I. When a tour reaches a segment s in I, then it makes a separating detour that separates all of the segments in $N(s)$. The length of a tour is bounded by $O(L^2)$ since all of the segments in $N(s)$ is within a neighborhood of radius $2r$ of s and by a packing argument, there are $O(L^2)$ possible points in such a neighborhood. In the worst case, the separating tour visits every point in the neighborhood and includes or excludes each point as required. Note that the detour must exclude segments that are not in $N(s)$.

Theorem 5. *The path our algorithm produces is a $O(L^2)$ approximation of the optimal minimum perimeter separator.*

Proof. Let T be the length of the path our algorithm produces, let OPT be the length of the optimal solution and let $TSPN(I)$ be the length of the TSPN path on I. Our path is a separating tour because every segment is separated by the tour within its neighborhood and excluded by the tour everywhere else. We claim the length of such a tour is bounded above by $TSPN(I)+O(|I|L^2)$. Since the TSPN path of I must enter and exit the neighborhood of every segment in I, $TSPN(I) = \Omega(|I|)$. Therefore $T = O(L^2 \cdot TSPN(I))$. Since $OPT = \Omega(TSPN(I))$ and $T \geq OPT$, then $T = O(L^2 \cdot OPT)$.

3.5 The General Case

For general pairs of points in the plane, we observe that an $O(\sqrt{n})$-approximation follows from known results on the Euclidean TSP in the plane. Specifically, we first compute a minimum-size square, Q, that contains at least one point of each pair. (This is easily computed, since the n point pairs determine only $O(n^3)$ combinatorially distinct squares.) Now, within Q, we compute an approximate TSP tour T (using any constant-factor approximation method for TSP) on the points that are inside (or on the boundary of) Q, making sure the approximate tour is a simple polygon. We obtain a valid separating cycle for the input pairs as follows: Consider traversing T, starting from an arbitrary point. Each time we reach a point along this traversal, we either make a slight detour to include it (if it is the first time we have encountered a point from this pair), or make a slight detour to exclude it (if it is the second encounter with a pair). In this way, we obtain a valid separating cycle just slightly longer than T. By classic results on the Euclidean TSP (see, e.g., Karloff [9]), we know that the length of T is at most $O(|Q|\sqrt{n})$, where $|Q|$ is the side length of the square Q. Since we know that $\Omega(|Q|)$ is a lower bound on the length of an optimal separating cycle, we have shown that in polynomial time one can obtain an $O(\sqrt{n})$-approximation for the general case of our problem.

3.6 Separating Subdivision Problem

We consider now a different version of the separating cycle problem – the *separating subdivision problem*, in which the goal is to compute an embedded planar graph of minimum length such that every input pair has its points in different faces of the subdivision. We define the length of the graph to be the sum of the lengths of all of its edges. We give an $O(\log n)$-approximation algorithm. We outline the approach, deferring details to the full paper. We argue that an optimal subdivision, S, can be converted to a special (recursive) "guillotine" structure, increasing its length by a factor $O(\log n)$; then, we show that an optimal solution among guillotine structures can be computed in polynomial time, using dynamic programming. The conversion goes as follows. First, increasing the total edge length of S by at most a constant factor, we can convert its faces to all be

rectilinear: we enclose S with its bounding box, and replace each face of S with a rectilinear polygon, with axis-parallel edges that lie on the grid induced by the input point pairs, while keeping all points within their respective faces. Then, we partition each simple rectilinear face into rectangles, adding axis-parallel chords that lie on the grid; this causes the total edge length to go up by a factor $O(\log n)$; see [10]. Then, using the charging scheme of [10], we know that we can convert the resulting *rectangular subdivision* to a *guillotine rectangular subdivision*, in which one can recursively partition the subdivision using axis-parallel "guillotine" cuts that do not enter the interior of rectangular faces. Optimizing the length of a guillotine rectangular subdivision is done with dynamic programming, in which subproblems are axis-aligned rectangles all of whose boundary is (by definition) included in the edge set of the subdivision. This implies that any input pair of points that "straddles" the boundary of a subproblem, with one point inside, one point outside, is already satisfied automatically with respect to pair separation (the points lie in different faces/rectangles). This means that the subproblem is only responsible for the separation of the point pairs both of whose points lie within the defining rectangle of the subproblem. The algorithm computes a minimum-length guillotine rectangular subdivision, separating all point pairs. Since an optimal solution can be converted to the class of guillotine rectangular subdivisions at a lengthening factor $O(\log n)$, we obtain the claimed approximation.

4 Conclusion and Future Work

The shortest separating cycle is a new variant of the TSP family that has not been studied before. This paper provides the first set of hardness bounds and a number of approximation algorithms under different settings. The gap for the approximation ratios and hardness results is still big and narrowing or closing the gap is the obvious future work.

Acknowledgement.

E. Arkin and J. Mitchell acknowledge support from NSF (CCF-1526406). J. Gao and J. Zeng acknowledge support from AFOSR (FA9550-14-1-0193) and NSF (CNS-1217823, DMS-1418255, CCF-1535900).

References

1. E. M. Arkin and R. Hassin. Approximation algorithms for the geometric covering salesman problem. *Discrete Appl. Math.*, 55(3):197–218, Dec. 1994.
2. S. Arora. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *J. ACM*, 45(5):753–782, sep 1998.
3. S. Arora and K. Chang. Approximation schemes for degree-restricted MST and redblue separation problems. *Algorithmica*, 40(3):189–210, 2004.

4. T. H. Chan and S. H. Jiang. Reducing curse of dimensionality: Improved PTAS for TSP (with neighborhoods) in doubling metrics. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 754–765, 2016.
5. N. Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical Report 388, Graduate School of Industrial Administration, Carnegie Mellon University, 1976.
6. A. Dumitrescu and J. S. Mitchell. Approximation algorithms for TSP with neighborhoods in the plane. *Journal of Algorithms*, 48(1):135 – 159, 2003. Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms.
7. M. R. Garey, R. L. Graham, and D. S. Johnson. Some NP-complete geometric problems. In *Proceedings of the Eighth Annual ACM Symposium on Theory of Computing*, STOC '76, pages 10–22, New York, NY, USA, 1976. ACM.
8. J. Gudmundsson and C. Levcopoulos. Hardness result for TSP with neighborhoods. Technical report, Technical Report LU-CS-TR:2000-216, Department of Computer Science, Lund University, Sweden, 2000.
9. H. J. Karloff. How long can a Euclidean traveling salesman tour be? *SIAM Journal on Discrete Mathematics*, 2(1):91–99, 1989.
10. C. Mata and J. S. B. Mitchell. Approximation algorithms for geometric tour and network design problems. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 360–369, 1995.
11. C. S. Mata and J. S. B. Mitchell. Approximation algorithms for geometric tour and network design problems (extended abstract). In *Proceedings of the Eleventh Annual Symposium on Computational Geometry*, SCG '95, pages 360–369, New York, NY, USA, 1995. ACM.
12. J. S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems. *SIAM Journal on Computing*, 28(4):1298–1309, 1999.
13. J. S. B. Mitchell. A PTAS for TSP with neighborhoods among fat regions in the plane. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 11–18, 2007.
14. J. S. B. Mitchell. A constant-factor approximation algorithm for TSP with pairwise-disjoint connected neighborhoods in the plane. In *Proc. 26th Annual ACM Symposium on Computational Geometry*, pages 183–191, 2010.
15. C. H. Papadimitriou. The Euclidean travelling salesman problem is NP-complete. *Theoretical Computer Science*, 4(3):237 – 244, 1977.
16. S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker. GHT: A geographic hash table for data-centric storage in sensornets. In *Proc. 1st ACM Workshop on Wireless Sensor Networks ands Applications*, pages 78–87, 2002.
17. S. Safra and O. Schwartz. On the complexity of approximating TSP with neighborhoods and related problems. *Computational Complexity*, 14(4):281–307, 2006.
18. R. Sarkar, X. Zhu, and J. Gao. Double rulings for information brokerage in sensor networks. *IEEE/ACM Trans. Netw.*, 17(6):1902–1915, Dec. 2009.
19. P. Slavík. The errand scheduling problem. Technical report 97-2, Department of Computer Science, SUNY, Buffalo, 1997.