A CONSTANT-FACTOR APPROXIMATION ALGORITHM FOR THE GEOMETRIC K-MST PROBLEM IN THE PLANE

JOSEPH S. B. MITCHELL*, AVRIM BLUM†, PRASAD CHALASANI‡, AND SANTOSH VEMPALA§

Abstract. We show that any rectilinear polygonal subdivision in the plane can be converted into a "guillotine" subdivision whose length is at most twice that of the original subdivision. "Guillotine" subdivisions have a simple recursive structure that allows one to search for "optimal" such subdivisions in polynomial time, using dynamic programming. In particular, a consequence of our main theorem is a very simple proof that the k-MST problem in the plane has a constant factor polynomial-time approximation algorithm: We obtain a factor of 2 (resp., 3) for the L_1 metric, and a factor of $2\sqrt{2}$ (resp., 3.266) for the L_2 (Euclidean) metric in the case in which Steiner points are allowed (resp., not allowed).

Key words. minimum spanning trees, k-MST, guillotine subdivisions, quota traveling salesman problem, prize-collecting salesman, and bank robber (orienteering) problem, network optimization, computational geometry, dynamic programming, approximation algorithms polynomial

AMS subject classifications. 68Q25, 68R10, 68U05

1. Introduction. We introduce a new technique that can be used to obtain simple approximation algorithms for geometric network design problems. The method is based on the concept of a "guillotine subdivision". Roughly speaking, a "guillotine subdivision" is a rectilinear polygonal subdivision with the property that there exists a horizontal or vertical line (a "cut") whose intersection with the edge set is *connected* and the subdivisions on either side of the line are also guillotine. The connectedness property allows one to apply dynamic programming to optimize over guillotine subdivisions, as there is a succinct specification of how the subdivision interacts with the "cuts" that make up the boundary of a rectangle that specifies a "subproblem" of the dynamic program.

Key to our method is a theorem showing that any rectilinear polygonal subdivision can be converted into a guillotine subdivision by adding a set of edges whose total length is small (at most that of the original subdivision).

To illustrate the power of the method, we show how it can be used to give a very simple constant-factor approximation algorithm for the geometric k-MST problem, obtaining a substantially better factor than previously known. We also apply it to some related problems (the "quota TSP", "prize-collecting salesman", and "bank robber (orienteering)" problems).

A Motivating Application. A special case of the "quota TSP" problem is the following: You are a salesman who must sell k items. You can sell one item in each of n cities ($n \geq k$). You want to find a shortest tour that visits at least k cities, so that you can sell your quota of k items. A solution (exact or approximate) to the k-MST

*Department of Applied Mathematics and Statistics, State University of New York, Stony Brook, NY 11794-3600. (jsbm@ams.sunysb.edu). Supported in part by Hughes Research Laboratories and NSF Grants CCR-9204585 and CCR-9504192.

†School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213-3891. (avrim@cs.cmu.edu). Supported in part by NSF National Young Investigator grant CCR-9357793 and a Sloan Foundation Research Fellowship.

‡Los Alamos National Laboratory, Los Alamos NM 87544. (chal@lanl.gov).

§School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213-3891. (svempala@cs.cmu.edu). Supported in part by NSF National Young Investigator grant CCR-9357793.

problem on the n cities immediately gives an approximation to the desired optimal tour (simply by doubling the tree, in the usual manner).

Related Work. In the minimum-weight k-tree, or k-MST problem, we are given a graph on n vertices with nonnegative distances on the edges, and an integer $k \leq n$, and our goal is to find a tree of least total weight that spans some subset of k vertices. The k-MST problem was introduced independently by Fischetti et al. [11], Zelikovsky and Lozevanu [22], and Ravi et al. [21]. In those papers, the problem is shown to be NP-complete, and Ravi et al. give an approximation algorithm with factor $O(\sqrt{k})$. Algorithms with improved approximation factors have since been discovered: Awerbuch et al. [2] obtain factor $O(\log^2 k)$, and Rajagopalan and Vazirani [20] obtain $O(\log k)$. In the time since our work first appeared (in [6] and [18]), there have been further improvements in the approximation factors: Blum, Ravi, and Vempala [7] obtain a factor of 17, and, most recently, Garg [12] obtains a factor of 3. Cheung and Kumar [8] have also considered the problem, which they call the "quorum-cast" problem and which arises in communication networks. Note that if $k = n$, the the k-MST problem is simply the usual minimum spanning tree problem, which has efficient (polynomial-time) exact solutions; thus, the complexity of the k-MST problem arises from the fact that we must find *which* k vertices to connect with a minimum spanning tree.

In the *geometric* k-MST problem, the underlying graph is the complete graph induced by a set of points in the plane, with distances between pairs of points determined by the underlying metric space (typically, this will be Euclidean (L_2) or L_1). Specifically, we are given a set P of n points in the plane, and an integer $k \leq n$, and we are to find a subset of k points of P that has the shortest minimum spanning tree. The problem is NP-hard. Ravi et al. [21] give an approximation algorithm with ratio $O(k^{1/4})$, which was quickly improved to a factor of $O(\log k)$ by Garg and Hochbaum [13] and Mata and Mitchell [17]. Eppstein [10] has improved the approximation ratio to $O(\log k/\log\log n)$, and has given general techniques to improve the running times (as a function of n) of existing algorithms; further, he shows that the exact k-MST problem can be solved in time $2^{O(k\log k)}n + O(n\log n)$, which is simply $O(n\log n)$ for fixed k.

Note that, up to small constant factors in the approximation ratio, the k-MST problem is equivalent to the problem of finding a shortest path or shortest tour that visits k points (the k-TSP problem) or a shortest Steiner tree connecting k points.

Our Contribution. Our result is a simple proof of an $O(1)$ approximation ratio for the geometric k-MST problem. We obtain a factor of 2 (resp., 3) (for the L_1 metric) or $2\sqrt{2}$ (resp., 3.266) (for the Euclidean metric) in the case that Steiner points are (resp., are not) allowed. Further, we expect that our guillotine subdivision results may yield similar improvements and simplifications to approximation algorithms for other geometric network design problems. In Section 6, we mention a few applications of our method to some problems that are related to the k-MST.

This paper represents the contributions from two manuscripts: the work of Blum, Chalasani, and Vempala [6], based on the notion of a "division tree", and the improvement and simplification to it given by Mitchell [18], based on the notion of "guillotine subdivisions". We will briefly describe the original "division tree" method of Blum et al., but will only give details of the simpler method of Mitchell [18].

2. Division Trees. In this section we define *division trees*, and describe a simple dynamic programming algorithm based on this notion that achieves a constant factor approximation to the k-MST problem. The proof of the approximation guarantee appears in [6]. We do not present the proof here because in the next section we will describe a more powerful algorithm that achieves a better constant factor, and for which the proof is significantly simpler.

To define a division tree, we assume for convenience that no two points lie on the same horizontal or vertical line. In this case, we say that a spanning tree T for a set of points is a *Division Tree* if T satisfies the following recursive property:

> There exists some point r (the "root") such that either the vertical or the horizontal line through r splits T into two division trees. More precisely, we require both that (A) this line does not intersect any edges of T, and (B) the trees T_1 and T_2 induced by the points on either side of the line *including* r should be division trees. For the base case, if there are just two points, then the single edge is a division tree.

Given any set of n points, the following simple dynamic programming algorithm finds the subset of k points having a division tree of minimum weight.

The algorithm is most easily viewed in a recursive "memoizing" form. It returns both the desired set of k points and the *cost* of the associated division tree. The algorithm takes as input a set of points P, an integer k, and also up to four additional constraints. For each of the 4 sides of the bounding box of P the algorithm may be told that the point on that bounding side is "required" and must be in any set of k points the algorithm produces. At the outer loop there are no required points. Given these inputs, the algorithm considers each vertical and horizontal line that passes through some point in P and does not coincide with an edge of the bounding box. For a given such line—let p be the point in P that the line passes through—the algorithm constructs the bounding boxes B_1 and B_2 of the points on the two sides, considering p to be on both sides. It then calls itself recursively $k-1$ times for each of the two boxes B_i: in each call passing down the set of points in B_i, a new integer $k' \in [2, k]$, and the set of required points it was originally given (only considering those that lie in the box B_i) *including* the new point p. Once the algorithm receives its $k-1$ answers from each side, it simply compares to find the pair $\langle k', k-k'+1\rangle$ whose costs sum to the least amount (the reason for the "+1" is that point p lies on both sides). In the base case, $k=2$, the algorithm just returns the cost of the single edge.

Because there are at most n^4 different bounding boxes, k different possibilities for the desired number of points, and 16 different settings for the "required points", the memoized procedure (or equivalently, dynamic program) will run in polynomial time. Also, it is not too hard to see that this algorithm finds the set of k points with the lightest division tree. What is shown in [6] is that for any set of points, the division tree of minimum weight is only a constant factor more costly than the minimum spanning tree.

3. Guillotine Subdivisions. We now turn to guillotine subdivisions, and our main theorem that any rectilinear subdivision of the plane can be approximated by one that is guillotine.

Consider a rectilinear polygonal subdivision S that is induced by a finite set of noncrossing horizontal and vertical (closed) line segments in the plane, whose union, E, comprises the edges of S. We can assume (without loss of generality) that S is restricted to the unit square, B (i.e., $E \subset int(B)$). Then, each facet (2-face) is a bounded rectilinear polygon, possibly with holes. The *length* of S is the sum of the lengths of the edges of S.

A closed, axis-aligned rectangle W is a *window* if $W \subseteq B$. In the following definitions, we fix attention on a given window, W.

A line ℓ is a *cut* for E with respect to W if $\ell \cap int(W) \neq \emptyset$. The intersection $\ell \cap (E \cap int(W))$ of a cut ℓ with $E \cap int(W)$, the *restriction* of E to the window W, consists of a discrete (possibly empty) set of subsegments of ℓ. (Some of these "segments" may be points, where ℓ crosses an edge.) The endpoints of these subsegments are called the *endpoints along* ℓ (with respect to W). (The two points where ℓ crosses the boundary of W are not considered to be endpoints along ℓ.) Let ξ be the number of endpoints along ℓ, and let the points be denoted by $p_1, \ldots, p_\xi$, in order along ℓ.

We define the *span* $\sigma(\ell)$ of ℓ (with respect to W) as follows. If $\xi = 0$, then $\sigma(\ell) = \emptyset$; otherwise, $\sigma(\ell)$ is defined to be the (possibly zero-length) line segment, $p_1 p_\xi$.

A line ℓ is a *perfect cut with respect to* W if $\sigma(\ell) \subseteq E$ (which implies that $\xi = 2$, or $\xi = 1$ in case $\sigma(\ell)$ is a single point).

Finally, we say that S is a *guillotine subdivision with respect to window* W if either (1) $E \cap int(W) = \emptyset$; or (2) there exists a perfect cut, ℓ, with respect to W, such that S is guillotine with respect to windows $W \cap H^+$ *and* $W \cap H^-$, where H^+, H^- are the closed halfplanes induced by ℓ. We say that S is a *guillotine subdivision* if S is guillotine with respect to the unit square, B.

See Figure 3.1 for an example of a guillotine subdivision, where we illustrate the entire tree of perfect cuts. (Each perfect cut is indicated with a small arrow.)

Note that, in contrast with guillotine *rectangular* subdivisions (see [9, 17]), the guillotine subdivisions we study here are not restricted to have rectangular faces; rather, the faces of a guillotine subdivision are rectilinear polygons. In fact, it is precisely this distinction that permits us to get constant factor approximations, while the previous method of [17] obtained logarithmic factors. For example, in order to transform a "staircase" (rectilinear) polygon into a guillotine rectangular subdivision, we must increase its total edge length by a factor of $\Omega(\log n)$; in contrast, a staircase polygon is already a guillotine subdivision, according to our definition.

4. The Main Theorem. We now show that any rectilinear subdivision can be converted into a guillotine subdivision, without increasing its length by much (at most doubling it). Our proof is inspired by the proof in [9] that any subdivision of a box (in $\Re^2$) into *rectangles* can be converted into a "guillotine" rectangular subdivision of at most twice the length, by adding a new set of edges whose total length is small (charged off to the original edges of the subdivision).

THEOREM 4.1. *Let S be a rectilinear subdivision of length L with edge set E. Then, there exists a guillotine subdivision, S_G, of length at most $2L$ whose edge set, E_G, contains E.*

Proof. We will convert S into a guillotine subdivision S_G by adding to E a new set of horizontal or vertical edges whose total length is at most L. The construction is recursive; at each stage, we show that there exists a cut, ℓ, with respect to the current window W (which initially is the box B), such that we can afford to add the span $\sigma(\ell)$ to E, while appropriately charging off the length of $\sigma(\ell)$. (Once we add $\sigma(\ell)$ to E, ℓ becomes a perfect cut with respect to W.)

In fact, we will restrict ourselves to a special discrete set of horizontal or vertical cuts, namely those determined by the x- or y-coordinates of original vertices V of the subdivision, or by the midpoints between consecutive x- or y-coordinates of V.

First, note that if a perfect cut (with respect to W) exists, then we can simply use it and proceed recursively on each side of the cut. Thus, we assume that no perfect

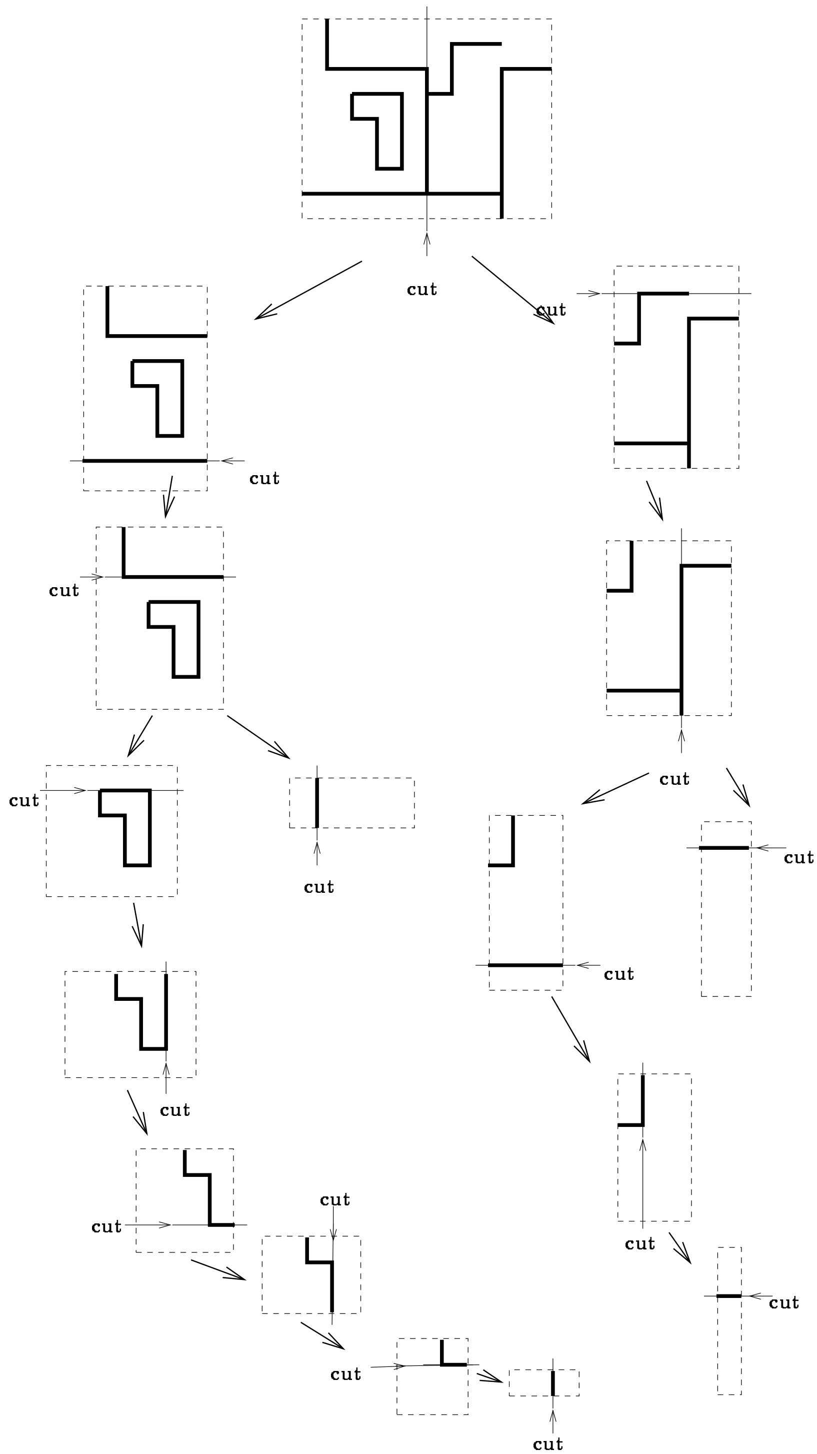

FIG. 3.1. *An example of a guillotine subdivision. Each perfect cut is indicated with a small arrow; bounding boxes (windows) are indicated with dashed rectangles.*

cut exists with respect to a given window, W.

We say that a point p on a cut ℓ is ***dark with respect to*** ℓ ***and*** W if, along $\ell^{\perp} \cap int(W)$, there is at least one endpoint (strictly) on each side of p, where $\ell^{\perp}$ is the line through p and perpendicular to ℓ.[1] We say that a subsegment of ℓ is ***dark*** (with respect to W) if all points of the segment are dark with respect to ℓ and W.

The important property of dark points along ℓ is the following: Assume, without loss of generality, that ℓ is horizontal. Then, if all points on subsegment pq of ℓ are dark, then we can charge the length of pq off to the bottoms of the subsegments, $E^+ \subseteq E$, of edges that lie above pq and are vertically visible to pq, and the tops of the subsegments, $E^- \subseteq E$, of edges that lie below pq and are vertically visible to pq (since we know that there is at least one edge "blocking" each point of pq from the top or bottom of W). We charge pq's length half to E^+ (charging E^+ *from below*, with $\frac{1}{2}$ units of charge) and half to E^- (charging E^- *from above*, with $\frac{1}{2}$ units of charge). In Figure 4.1 we illustrate how a dark subsegment, pq, has its length charged off.

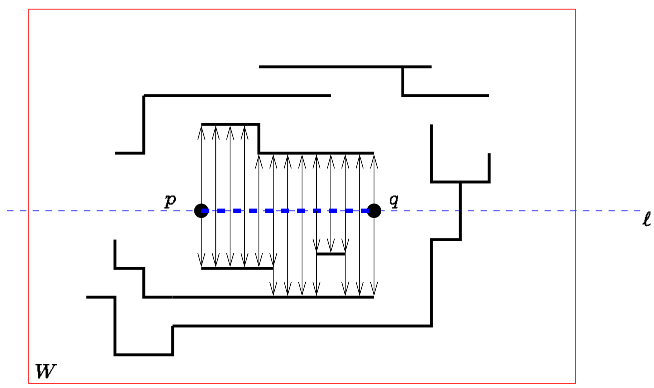

FIG. 4.1. *Subsegment $pq \subset \ell$ is dark; its length is charged to the subsegments of E that lie above or below.*

We call a cut ℓ *favorable* if the dark portion of ℓ is at least as long as the span, $\sigma(\ell)$. Lemma 4.2 below shows that a favorable cut always exists (even one in the special discrete set). For a favorable cut ℓ, we add its span to the edge set (charging off its length, as above) and recurse on each side of the cut in the two new windows. After a portion of E has been charged on one side, due to a cut ℓ, it will be vertically visible to the boundary of the windows on either side of ℓ, and, hence, will be vertically visible to the boundary of any future windows, found deeper in the recursion, that contain the portion. Thus, *no portion of E will ever be charged more than once from each side* (top and bottom), so no portion of E will ever pay more than its total length in charge ($\frac{1}{2}$ from each side). Also, the new edges added (the spans $\sigma(\ell)$) are never themselves charged, since they lie on window boundaries and cannot therefore serve to make a portion of some future cut dark.

Note too that, in order for a cut ℓ to be favorable, but not perfect, there must be at least one segment of E parallel to ℓ in each of the two open halfplanes induced by ℓ; thus, the recursion must terminate in a finite number of steps.

[1] We can think of the edges E as being "walls" that block light; then, p on a line ℓ is dark if p is not illuminated when light is shone in from the boundary of W, along the direction of $\ell^{\perp}$.

Since the total length of all spans for all favorable cuts is at most L, and the total length of all spans for all perfect cuts is at most L, we are done. □

We now prove our key lemma, which guarantees the existence of a favorable cut when there is no perfect cut. The proof of the lemma uses a particularly simple argument, based on elementary calculus (reversing the order of integration):

LEMMA 4.2. *For any subdivision S, and any window W, there must be a favorable cut.*

Proof. We show that there must be a favorable cut that is either horizontal or vertical.

Let $f(x)$ denote the length of the span (with respect to W) of the vertical line through x. Then, $\int_0^1 f(x)dx$ is simply the area, A_x, of the (x-monotone) region R_x of points of B that are dark with respect to horizontal cuts. Similarly, define $g(y)$ to be the length of the span of the horizontal line through y, and let $A_y = \int_0^1 g(y)dy$.

Assume, without loss of generality, that $A_x \geq A_y$. We claim that there exists a horizontal favorable cut; i.e., we claim that there exists a horizontal cut, ℓ, such that the length of its dark portion is at least as large as the length of its span, $\sigma(\ell)$. To see this, note that A_x can be computed by switching the order of integration, "slicing" the region R_x horizontally, rather than vertically; i.e., $A_x = \int_0^1 h(y)dy$, where $h(y)$ is the length of the intersection of R_x with a horizontal line through y; i.e., $h(y)$ is the length of the dark portion of the horizontal line through y. Thus, since $A_x \geq A_y$, we get that $\int_0^1 h(y)dy \geq \int_0^1 g(y)dy \geq 0$. Thus, it cannot be that for all values of $y \in [0,1]$, $h(y) < g(y)$, so there exists a $y = y^*$ for which $h(y^*) \geq g(y^*)$. The horizontal line through this y^* is a cut satisfying the claim of the lemma. (If, instead, we had $A_x \leq A_y$, then we would get a *vertical* cut satisfying the claim.)

Finally, we note that, in the rectilinear case, f, g, and h are piecewise-constant, with discontinuities corresponding to vertices V of S. Then, we can always select y^* to be at a discontinuity or at a midpoint between two discontinuities. □

Remark.. It is interesting to consider whether or not the factor of 2 in the theorem can be improved. We have not been able to find an example of a subdivision that cannot be made into a guillotine subdivision by the addition of edges whose total length is less than *half* that of the original subdivision. We conjecture that the factor 2 can be improved to a factor of 3/2.

5. An Application to the k-MST. One application of our theorem is that it yields an algorithm along with a simple proof that it achieves a (small) constant-factor approximation of the geometric k-MST problem.

COROLLARY 5.1. *The geometric k-MST problem has a polynomial-time approximation algorithm, with approximation factor 2 (allowing Steiner points) or 3 (not allowing Steiner points), in the L_1 metric. In the Euclidean metric, the approximation factors become $2\sqrt{2}$ (Steiner) or $4\sqrt{6}/3 = 3.266$ (non-Steiner).*

Proof. Let P be a set of n points in the plane. Assume that no two points of P lie on a common vertical or horizontal line (otherwise, we can perturb the points or slightly rotate the coordinate system). Let T_R be a minimum-length rectilinear Steiner k-MST for P, and let L_R^* denote its total (Euclidean) length. We can assume that T_R lies on the grid of horizontal or vertical lines through P, since T_R can easily be modified to lie on the grid, without increasing its overall length. Clearly, the length, L_R^*, of T_R is the optimal length for the L_1 Steiner k-MST and is at most $\sqrt{2}$ times the length, L^*, of an optimal Euclidean Steiner k-MST.

By Theorem 4.1, there exists a guillotine subdivision, S_G, with edge set E_G, of length at most $2L_R^*$ (at most $2\sqrt{2}L^*$), such that $T_R \subseteq E_G$. Thus, there exists a perfect cut ℓ (inducing closed halfplanes $\bar{H}^+$ and $\bar{H}^-$) for E_G, such that $\ell \cap E_G$ is connected (a segment). When we select a favorable cut, whose existence is shown in Lemma 4.2, we can always select the cut either to pass through a point of P or to pass through, say, the midpoint of some x- or y-interval determined by consecutive coordinates of points of P; thus, the cuts will have coordinates from a *discrete* set determined by P. Proceeding recursively on each side of ℓ, we can build a tree $\mathcal{T}$ of perfect cuts.

The bounding boxes corresponding to each node of this tree have the special property that the intersection of E_G with each side of a box is *connected*; thus, we can partition the problem into subproblems, each having *constant*-size (discrete) specification, and can easily apply dynamic programming to search for an optimal guillotine subdivision that visits k of the points P and has a connected set of edges E. This yields a (rectilinear) Steiner tree connecting k points, of total Euclidean length at most $2L_R^*$. If our goal is a Steiner k-MST approximation, then we are done. Otherwise, at the end we compute and output a minimum spanning tree for this subset of k points. The worst-case length of the final tree is obtained by multiplying by the Steiner ratio ($3/2$ for L_1, $2\sqrt{3}/3$ for L_2). Thus, we get an approximation factor of $2 \cdot (3/2) = 3$ (for L_1 metric) or of $2\sqrt{2} \cdot (2\sqrt{3}/3) = 4\sqrt{6}/3 = 3.266$ (for L_2 metric). We give details of the dynamic programming algorithm below. □

A Dynamic Programming Algorithm. Let $x_1 < x_2 < \cdots < x_{2n-1}$ (resp., $y_1 < y_2 < \cdots < y_{2n-1}$) denote the sorted x (resp., y) coordinates of the n points P, as well as the $n-1$ midpoints of the intervals determined by these coordinates. We now give a dynamic programming algorithm, which is based on solving the following subproblems:

Input:

1. An integer $k' \leq k$;
2. a rectangle $R(i,i',j,j')$, defined by x_i, $x_{i'}$ ($x_{i'} > x_i$), y_j, and $y_{j'}$ ($y_{j'} > y_j$);
3. four "boundary segments", σ_l, σ_r, σ_b, and σ_t, which are (possibly empty or zero-length) subsegments of the four sides (left, right, bottom, top) of $R(i,i',j,j')$, each of which has endpoints determined by coordinates x_i, y_j; and,
4. a partition, $\mathcal{P}$, of the set $\{\sigma_l, \sigma_r, \sigma_b, \sigma_t\}$ of boundary segments.

Objective: Compute a minimum-length guillotine subdivision, S_G^*, determined by some set E_G^* of horizontal or vertical line segments not lying on the boundary of rectangle $R(i,i',j,j')$, such that E_G^* covers at least k' points of P (interior to $R(i,i',j,j')$) and the edges E_G^* connect the boundary segments, according to the partition $\mathcal{P}$.

Note that there are $O(k \cdot n^4 \cdot (n^2)^4) = O(kn^{12})$ possible inputs (subproblems). Note too that an optimal solution, S_G^* will necessarily be a forest, since any cycle that is formed can be broken without violating the connectivity requirements (given by $\mathcal{P}$).

The optimal value, V, of the above problem is 0 if $k' = 0$ and all connections among boundary segments specified by $\mathcal{P}$ are vacuously satisfied (i.e., the boundary segments that need to be connected are already connected at their endpoints (corners of the boundary box)). Otherwise, we can compute the value V recursively, by adding the values of the two subproblems obtained by splitting the problem, and optimizing over all choices associated with a split:

1. $O(k)$ choices of how to partition k' among the two new subproblems;

2. $O(n)$ choices of a cut by a horizontal or vertical line (determined by some x_i or y_j);
3. $O(n^2)$ choices of new boundary segment, σ, on the cut; and,
4. $O(1)$ choices of partitions for the two sets of boundary segments on the two new subrectangles determined by the cut, subject to these partitions being consistent with the partition $\mathcal{P}$.

The polynomial time bound for solving the above recursions has a rather high exponent — $O(n^{15}k^2)$. One approach to improving this is to apply the following lemma of Eppstein [10], which is obtained by doing a simple nearest-neighbor clustering:

LEMMA 5.2 (Eppstein 95). *If we have a time bound $T(n,k)$ for an exact or approximate geometric k-MST problem, we can solve the same problem in time $O(n \log n + nT(k^2,k)/k^2)$.*

A direct application of the above yields a time bound of $O(n \log n + nk^{30})$, which is an improvement when k is small compared with n.

Remark:. Awerbuch et al. [2] discuss both "rooted" and "unrooted" versions of the k-MST problem. In the "rooted" version of the problem, we are given a specified point, r, and we must use r as one of the k points in the MST. They prove that their $O(\log^2 k)$-approximation method applies to both problems, but the approximation ratio may increase by 1 for the rooted case. It is easy to see that our dynamic programming algorithm allows us also to obtain a rooted solution that is within the *same* factor of optimal as in the unrooted case, simply by specifying the appropriate constraint in the input to the dynamic program.

6. Other Applications. Our methods also apply to three other problems that are related to the k-MST, and for which approximation algorithms for the graph versions have been given by Awerbuch et al. [2]:

1. In the *quota-driven salesman* problem, each point of P has an associated integral value w_i, and a salesman has a given integer quota, R. The objective is to find a shortest possible route (or tour) such that the sum of the values for the cities visited is at least R. It is immediate that our k-MST approximation gives an approximation for this problem too: Simply replace each point of P by w_i copies of itself, at the same location in the plane. Now, simply compute an approximate solution to the k-MST, with $k = R$, and then double the tree to obtain a path or a tour.
   This algorithm runs in time polynomial in n and R. One open problem is to extend the algorithm to run in time polynomial in n and $\log R$, which holds for the algorithm of Awerbuch et al. [2].
2. In the *prize collecting salesman* problem (or, PCTSP), as studied by Balas [3] (see also [4]), the setup is the same as in the quota-driven salesman problem, except that, in addition to "values" w_i, there are non-negative penalties associated with each point of P, and the objective function is now to minimize the sum of the distances traveled *plus* the sum of the penalties on the points *not* visited, subject to satisfying the quota R. (Thus, if all penalties are 0, we simply get the quota-driven salesman problem.) As mentioned in [2], an approximation of PCTSP follows immediately from concatenating a tour obtained for the quota-driven salesman, with the 2-approximation tour given by the algorithm of Goemans and Williamson [15] (which considers the effect of penalties, but does not use the quota constraint).
3. In the *bank robber (orienteering)* problem, we are faced with essentially the dual of the quota-driven salesman problem: Given a gas tank that allows one

to travel a distance d, maximize the total value R of all points visited. As in [2], we can obtain an approximation for this problem, based on "guessing" the value of R, running the approximation for the quota-driven salesman for quota R, breaking the path into subpaths of length $d/2$, and then picking the subpath of highest value.
This reduction holds only for the *unrooted* version of this problem. For the rooted version (one has a fixed root and a given distance d, and the goal is to visit as many points as possible and return to the root without running out of gas) it is unclear how to obtain any nontrivial approximation.

4. In the *minimum latency problem* (MLT). also known as the *deliveryman problem* and the *traveling repairman problem*, we are given a set of points and must find a tour that minimizes the sum of the "latencies" of all points, where the *latency* of a point p is the length of the tour from the starting point to the point p. (Thus, the latency of a point measures how long a job at that point must wait before being served by the repairman/deliveryman that is traveling along the tour.) Blum et al. [5] have given an approximation algorithm with a constant factor bound of 128; this bound has recently been improved by Goemans and Kleinberg [14] to 29. By a direct application of Theorem 2 of [5], which states that a c-approximation for the k-MST implies an $8c$-approximation for the MLP, we see that our results immediately imply a 24-approximation algorithm for the L_1 metric MLP and a 26.13-approximation algorithm for the L_2 metric MLP, for points in the plane.

In a recent application of our guillotine subdivision results, Mata and Mitchell [16] have obtained a constant-factor approximation algorithm for the following *red-blue separation* problem: Given n points in the plane, each colored red or blue, find a shortest simple polygon separating red from blue. This problem is known to be NP-hard, and previously had an $O(\log n)$ approximation algorithm [17].

7. Conclusion. In conclusion, we mention some of the exciting developments that have happened in the time since this paper was written. In the spring of 1996, S. Arora [1] announced a remarkable result — he had found a polynomial-time approximation scheme, (PTAS) for the Euclidean traveling salesperson problem (TSP), as well as other geometric optimization problems, such as the Steiner tree problem, the k-MST problem, etc.[2] Then, some weeks later, Mitchell [19] discovered that, in fact, a very minor variation of the method and proof given in his earlier work [18], and contained in this paper, also gives a particularly simple PTAS for geometric instances of the TSP, the k-MST, the Steiner tree problem, etc. All that must be modified is the definition of "span", from "span" to "m-span" (which links the mth endpoint with the mth-from-the-last endpoint along a cut ℓ), and, in the proof of Theorem 4.1, the notion of "darkness", from "darkness" to "m-darkness" (p is *m-dark* with respect to ℓ and W if, along $\ell^{\perp} \cap int(W)$, there are at least m endpoints (strictly) on each side of p). Here, we gave the case of $m = 1$. By allowing any positive integer m, the exact same proof goes through, resulting in the following extension to Theorem 4.1: *For any rectilinear subdivision S, with edge set E, of length L, and for any positive integer m, there exists an m-guillotine rectilinear subdivision, S_G, of length at most $(1 + \frac{1}{m})L$ whose edge set, E_G, contains E.* (The same proof also applies to general (non-rectilinear) subdivisions, resulting in an m-guillotine subdivision of length at

[2] This means that, for any fixed $\epsilon > 0$ there exists a polynomial-time algorithm that gets within a factor $(1 + \epsilon)$ of optimal.

most $(1 + \frac{\sqrt{2}}{m})L$.) See [19] for further details of the application of this extension.

Acknowledgments. We thank D. Eppstein, M. Held, S. Khuller and C. Mata for comments and suggestions that improved this paper.

REFERENCES

[1] S. Arora, *Polynomial time approximation schemes for Euclidean TSP and other geometric problems*, Manuscript, March 30, 1996. Appears in Proc. 37th Annu. IEEE Sympos. Found. Comput. Sci.(1996), pp. 2–12.

[2] B. Awerbuch, Y. Azar, A. Blum, and S. Vempala, *Improved approximation guarantees for minimum-weight k-trees and prize-collecting salesmen*, Proc. 27th Annu. ACM Sympos. Theory Comput. (1995), pp. 277–283.

[3] E. Balas, *The prize collecting traveling salesman problem*, Networks, 19 (1989), pp. 621–636.

[4] D. Bienstock, M. X. Goemans, D. Simchi-Levi, and D. Williamson, *A note on the prize collecting traveling salesman problem*, Math. Prog., 59 (1993), pp. 413–420.

[5] A. Blum, P. Chalasani, D. Coppersmith, B. Pulleyblank, P. Raghavan, and M. Sudan, *The minimum latency problem*, Proc. 26th Annu. ACM Sympos. Theory Comput. (1994), pp. 163–171.

[6] A. Blum, P. Chalasani, and S. Vempala, *A constant-factor approximation for the k-MST problem in the plane*, Proc. 27th Annu. ACM Sympos. Theory Comput. (1995), pp. 294–302.

[7] A. Blum, R. Ravi, and S. Vempala, *A constant-factor approximation algorithm for the k-MST problem*, Proc. 28th Annu. ACM Sympos. Theory Comput. (1996), pp. 442–448.

[8] S. Y. Cheung and A. Kumar, *Efficient quorumcast routing algorithms*, Proc. IEEE INFOCOM '94 Conference on Computer Communications, 2 (1994), pp. 840–847.

[9] D.-Z. Du, L.-Q. Pan, and M.-T. Shing, *Minimum edge length guillotine rectangular partition*, Report 02418-86, Math. Sci. Res. Inst., Univ. California, Berkeley, CA, 1986.

[10] D. Eppstein, *Faster geometric k-point MST approximation*, Tech. Report 95-13, Dept. Inform. Comput. Sci., Univ. California, Irvine, CA, 1995.

[11] M. Fischetti, H. W. Hamacher, K. Jørnsten, and F. Maffioli, *Weighted k-cardinality trees: complexity and polyhedral structure*, Networks, 24 (1994), pp. 11–21.

[12] N. Garg, *A 3-approximation for the minimum tree spanning k vertices*, Proc. 37th Annu. IEEE Sympos. Found. Comput. Sci.(1996), pp. 302–309.

[13] N. Garg and D. S. Hochbaum, *An $O(\log k)$ approximation algorithm for the k minimum spanning tree problem in the plane*, Proc. 26th Annu. ACM Sympos. Theory Comput. (1994), pp. 432–438.

[14] M. Goemans and J. Kleinberg, *An improved approximation ratio for the minimum latency problem*, Proc. 7th ACM-SIAM Sympos. Discrete Algorithms (1996), pp. 152–158.

[15] M. Goemans and D. Williamson, *General approximation technique for constrained forest problems*, Proc. 3rd ACM-SIAM Sympos. Discrete Algorithms (1992), pp. 307–315.

[16] C. Mata and J. S. B. Mitchell, *A constant factor approximation algorithm for the red-blue separation problem*, Manuscript, Dept. of Computer Science, SUNY Stony Brook, NY, 1995.

[17] C. Mata and J. S. B. Mitchell, *Approximation algorithms for geometric tour and network design problems*, Proc. 11th Annu. ACM Sympos. Comput. Geom. (1995), pp. 360–369.

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

[19] J. S. B. Mitchell, *Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems*, Manuscript, April 30, 1996. To appear, SIAM J. Comp.

[20] S. Rajagopalan and V. Vazirani, *Logarithmic approximation of minimum weight k trees*, Manuscript, 1995.

[21] R. Ravi, R. Sundaram, M. V. Marathe, D. J. Rosenkrantz, and S. S. Ravi, *Spanning trees short and small*, Proc. 5th ACM-SIAM Sympos. Discrete Algorithms (1994), pp. 546–555.

[22] A. Zelikovsky and D. Lozevanu, *Minimal and bounded trees*, Tezele Cong. XVIII Acad. Romano-Americane, Kishinev (1993), pp. 25–26.