Algorithmica (1989) 4: 569–583

Algorithmica

Approximate Minimum Weight Matching on Points in k-Dimensional Space[1]

Pravin M. Vaidya[2]

Abstract. We study the problem of finding a minimum weight complete matching in the complete graph on a set V of n points in k-dimensional space. The points are the vertices of the graph and the weight of an edge between any two points is the distance between the points under some L_q-metric. We give an $O((2c_q)^{1.5k}\varepsilon^{-1.5k}(\alpha(n,n))^{0.5}n^{1.5}(\log n)^{2.5})$ algorithm for finding an almost minimum weight complete matching in such a graph, where $c_q = 6k^{1/q}$ for the L_q-metric, α is the inverse Ackermann function, and $\varepsilon \le 1$. The weight of the complete matching obtained by our algorithm is guaranteed to be at most $(1+\varepsilon)$ times the weight of a minimum weight complete matching.

Key Words. Geometric matching, Approximation algorithms.

1. Introduction. Given a complete weighted undirected graph on a set of n vertices, a complete matching is a set of $n/2$ (n even) edges such that every vertex has exactly one edge in the matching incident on it. The weight or cost of a set of edges is the sum of the weights of the edges in the set and the weight of a graph is the weight of the set of its edges. A minimum weight complete matching (MWCM) is a complete matching that has the least weight among all the complete matchings. The problem of finding a minimum weight complete matching is a very-well-studied problem and an MWCM in a complete graph on n vertices may be found in $O(n^3)$ time [4], [5].

We study the problem of finding an MWCM in the complete graph on a set V of n points in k-dimensional space. The points are the vertices of this graph and the weight of an edge between any two points is the distance between the points under some L_q metric. Each point x is given as vector $(x_1, x_2, \ldots, x_k)$. The L_q distance between two points $x = (x_1, x_2, \ldots, x_k)$ and $y = (y_1, y_2, \ldots, y_k)$ is given by $\left(\sum_{i=1}^{k} |x_i - y_i|^q\right)^{1/q}$. (Note that the L_∞ distance between x and y is given by $\max_i |x_i - y_i|$.) We assume that the dimension k and the metric L_q are fixed. By a complete matching on V we mean a complete matching in the complete graph on V.

A complete graph induced by a set V of n points in k-dimensional space is entirely specified by n k-tuples of real numbers which give the locations of the vertices. Thus the problem of finding an MWCM on such a set of points V differs from the problem of finding an MWCM in a general complete graph in that the input size is $O(n)$ rather than $\Omega(n^2)$. The input is sparse because the edge weights

[1] This research was supported by a fellowship from the Shell Foundation.

[2] 2D-153, AT&T Bell Laboratories, 600 Mountain Avenue, Murray Hill, NJ 07974, USA.

Received September 1, 1986; revised January 5, 1988. Communicated by C. K. Wong.

are defined implicitly by the underlying geometry. It is interesting to investigate if the geometric nature of the graph can be exploited to obtain fast algorithms that find complete matchings whose weight is minimum or close to minimum. Several algorithms that are based on the underlying geometry may be found in [1], [8], and [11]. However, all the algorithms have the drawback that in the worst case the weight of the complete matching obtained may be far from minimum. We present an algorithm that finds an almost minimum weight complete matching on a set V of n points in k-dimensional space in $O((2c_q)^{1.5k}\varepsilon^{-1.5k}(\alpha(n,n))^{0.5}n^{1.5}(\log n)^{2.5})$ time, where $c_q = 6k^{1/q}$ for the L_q-metric, α is the inverse Ackermann function [12], and $\varepsilon \leq 1$. The weight of the complete matching obtained is guaranteed to be at most $(1+\varepsilon)$ times the weight of an MWCM on V. We also describe a heuristic that runs in $O(n(\log n)^3)$ time, and finds a complete matching on V whose weight is at most $3\log_3(1.5n)$ times the weight of an MWCM on V.

We let $M_{\text{opt}}(V)$ denote an MWCM on the finite set of points V, and let $M_{\text{opt}}(G)$ denote an MWCM in the graph G. We let $w(e)$ denote the weight of an edge e. We also let $w(E)$ denote the weight of the set of edges E, and let $w(G)$ denote the weight of graph G. An odd degree subgraph of G is a subgraph of G such that each vertex in G has odd degree in the subgraph. A minimum weight odd degree subgraph of G is an odd degree subgraph of G that has the least weight among all the odd degree subgraphs of G. We let $\psi_{\text{opt}}(G)$ denote a minimum weight odd degree subgraph of G. As the metric L_q is fixed we use distance for L_q distance and let $d(p, p')$ denote the distance between two points p and p'. For a set of points V, we let $d_{\min}(V)$ and $d_{\max}(V)$ respectively denote the minimum and the maximum distance between a pair of points in V. For sets of points V_1 and V_2, we let $d_{\min}(V_1, V_2)$ and $d_{\max}(V_1, V_2)$ respectively denote the minimum and the maximum distance between a point in V_1 and a point in V_2. A box is defined to be the product $J_1 \times J_2 \times \cdots \times J_k$ of k intervals, each interval being closed on the left and open on the right. Alternately, a box is the set of those points $x = (x_1, x_2, \ldots, x_k)$ such that x_i is in interval J_i (J_i being closed on the left and open on the right), $i = 1, 2, \ldots, k$. A box is cubical iff all the k intervals defining it have the same length, and the size of a cubical box is the length of each of the k intervals defining it.

2. An Overview. Let V be the given set of n points in k-dimensional space. Let us assume that $d_{\max}(V)/d_{\min}(V)$ is bounded by n^8. In Section 5 we describe how to reduce the given problem, in $O(n \log n)$ time, to a problem where the ratio $d_{\max}(V)/d_{\min}(V)$ is bounded by n^8. The algorithm for finding an almost minimum weight complete matching on such a set of points V proceeds as follows. Let ε be a parameter less than or equal to 1, and let c_q be a constant such that $c_q = 6k^{1/q}$ for the L_q-metric, $q = 1, 2, \ldots, \infty$.

1. From the set of points V extract a sparse graph $G = (V, E)$ such that $|E| = O((2c_q)^k \varepsilon^{-k} n \log n)$ and $w(\psi_{\text{opt}}(G)) \leq (1+\varepsilon) w(M_{\text{opt}}(V))$.
2. Find an odd degree subgraph Γ of G such that $w(\Gamma) \leq (1+1/n) w(\psi_{\text{opt}}(G))$.

3. Convert Γ into a complete matching M on V such that $w(M) \leq w(\Gamma)$. Then $w(M) \leq (1+1/n)(1+\varepsilon)w(M_{\text{opt}}(V))$.

In Section 4 we describe how to extract G from V in $O((2c_q)^k \varepsilon^{-k} n \log n)$ time.

In Section 6 we describe how to find an odd degree subgraph Γ of G, such that $w(\Gamma) \leq (1+1/n)w(\psi_{\text{opt}}(G))$, in $O((2c_q)^{1.5k} \varepsilon^{-1.5k} (\alpha(n, n))^{0.5} n^{1.5} (\log n)^{2.5})$ time. (α is the inverse Ackermann function [12].)

In Section 3 we show how to convert the odd degree subgraph Γ of G into a complete matching M on V, such that $w(M) \leq w(\Gamma)$, in $O(|V|+|E|)$ time. (The matching M is not necessarily a subgraph of Γ.)

Thus the entire process of finding a complete matching M on V such that

$$w(M) \leq (1+1/n)(1+\varepsilon)w(M_{\text{opt}}(V))$$

takes $O((2c_q)^{1.5k} \varepsilon^{-1.5k} (\alpha(n, n))^{0.5} n^{1.5} (\log n)^{2.5})$ time.

We note that the hypergreedy heuristic in [9] may be run on the graph G to obtain an odd degree subgraph Γ of G such that $w(\Gamma) \leq 2 \log_3(1.5n) w(\psi_{\text{opt}}(G))$. It is shown in [9] that given a graph $\hat{G} = (\hat{V}, \hat{E})$, the hypergreedy heuristic runs in $O(|\hat{E}|(\log|\hat{E}|)^2)$ time. Thus if in the above approximation algorithm we fix ε to be $\frac{1}{2}$, and utilize the hypergreedy heuristic to find an odd degree subgraph Γ of G in step 2, we get a procedure that runs in $O(n(\log n)^3)$ time, and finds a complete matching M on V such that $w(M) \leq 3 \log_3(1.5n) w(M_{\text{opt}}(V))$.

3. Converting an Odd Degree Subgraph of G into a Complete Matching on V. The odd degree subgraph Γ of $G = (V, E)$ is converted into a complete matching M on V as follows. We first find the connected components $\Gamma_1, \Gamma_2, \ldots, \Gamma_l$ of Γ. Each connected component must contain an even number of points (vertices) which may be seen as follows. The sum of the degrees of all the vertices in a connected component is an even number whereas the degree of each vertex is an odd number; so the number of vertices in each component must be even. Let $T_1, T_2, \ldots, T_l$ be spanning trees on the l connected components of Γ. Utilizing T_i we find a traveling salesman tour C_i of the points in Γ_i such that $w(C_i) \leq 2w(T_i)$. The tour C_i induces two complete matchings of the points in Γ_i and out of these two matchings we let M_i be the one of smaller weight (length). Then $M = \bigcup_{i=1}^{l} M_i$ is a complete matching on V and

$$w(M) = w\left(\bigcup_{i=1}^{l} M_i\right) \leq \sum_{i=1}^{l} w(T_i) \leq \sum_{i=1}^{l} w(\Gamma_i) \leq w(\Gamma).$$

The entire procedure for obtaining a matching M from an odd degree graph Γ may be implemented in time proportional to the number of edges in Γ.

4. Extracting Sparse Graph $G = (V, E)$. We are given a set V of n points in k-dimensional space satisfying the condition $(d_{\max}(V)/d_{\min}(V)) \leq n^8$. Let ε be a parameter less than or equal to 1, and let c_q be a constant defined as $c_q = 6k^{1/q}$

for the L_q-metric. We describe how to extract a sparse graph $G=(V, E)$ from V such that $|E|=O((2c_q)^k\varepsilon^{-k}n\log n)$ and $w(\psi_{\text{opt}}(G))\le(1+\varepsilon)w(M_{\text{opt}}(V))$.

Let g_0 be a smallest cube enclosing all the n points in V, and let L_0 be the length of a side of g_0. Let g_i be a grid that partitions g_0 into 2^{ki} identical cubical boxes, and let $\delta=\lceil(\log_2(2kn^8))\rceil$. Let B_i denote the set of those boxes (cubes) in g_i which contain a point in V, and let $B=\bigcup_{i=0}^{\delta}B_i$. We note that each box in B_δ contains exactly one point in V. Let $L_i=c_q\lceil\varepsilon^{-1}\rceil 2^{-i}L_0$. Let $r(b)$ denote the representative point in a box b.

Let Z denote the set of all the edges in the complete graph on V, and let

$$Z_i=\{(p_1, p_2): p_1\in b_1\cap V, p_2\in b_2\cap V, b_1\in B_i, b_2\in B_i, d_{\min}(b_1, b_2)\ge L_i/3\}.$$

The following two lemmas follow directly from the definitions.

LEMMA 1. *For $1\le i\le\delta$, if $b_1\in B_i$, $b_2\in B_i$, and $d_{\min}(b_1, b_2)\ge L_i/3$, then $d_{\max}(b_1)=d_{\max}(b_2)\le(\varepsilon/2)d_{\min}(b_1, b_2)$.*

LEMMA 2. $Z_0\subseteq Z_1\subseteq\cdots\subseteq Z_i\subseteq\cdots\subseteq Z_\delta\subseteq Z.$

LEMMA 3. *For $1\le i\le\delta$, if $(p_1, p_2)\in(Z_i-Z_{i-1})$, then p_1, p_2, are located in boxes b_1, b_2, which satisfy $b_1\in B_i$, $b_2\in B_i$, and $L_i/3\le d_{\min}(b_1, b_2)\le L_i$.*

PROOF. Suppose p_1, p_2 are located in boxes b_1, b_2 in B_i respectively. Since $(p_1, p_2)\in Z_i$, $d_{\min}(b_1, b_2)\ge L_i/3$. Let $b_1'\in B_{i-1}$, $b_2'\in B_{i-1}$, and let $b_1\subseteq b_1'$, $b_2\subseteq b_2'$. Since $(p_1, p_2)\in(Z_i-Z_{i-1})$, we have $d_{\min}(b_1', b_2')\le L_{i-1}/3$. Then

$$\begin{aligned} d_{\min}(b_1, b_2)&\le d_{\min}(b_1', b_2')+d_{\max}(b_1')/2+d_{\max}(b_2')/2\\ &\le L_{i-1}/3+k^{1/q}2^{-(i-1)}L_0\\ &\le L_i. \end{aligned}$$ □

We now give an algorithm to extract the sparse graph $G=(V, E)$ from the set of points V.

ALGORITHM *Sparse-Graph*

1. For each box $b\in B$, pick a representative point $r(b)$ from among the points in $b\cap V$.
2. Let $E=\bigcup_{i=1}^{\delta}E_i$ where

$$E_i=\{(r(b), p): b\in B_i, b'\in B_i, p\in b'\cap V, p\ne r(b), d_{\min}(b, b')\le L_i\}.$$

end *Sparse-Graph*

We first bound the number of edges in E. Assume $p \in b \cap V$ and $b \in B_i$. The edges in E_i that are incident to p connect p to representative points in boxes $b' \in B_i$ such that $d_{\min}(b, b') \le L_i$, and there are $O((2c_q)^k \varepsilon^{-k})$ such boxes in B_i. Thus each point has $O((2c_q)^k \varepsilon^{-k})$ edges in E_i incident on it and so $|E_i| = O((2c_q)^k \varepsilon^{-k} n)$. As $\delta = O(\log n)$ this gives

$$|E| \le \sum_{i=1}^{\delta} |E_i| = O((2c_q)^k \varepsilon^{-k} n \log n) = O(\varepsilon^{-k} n \log n) \quad \text{for fixed } k.$$

We now prove that $w(\psi_{\text{opt}}(G)) \le (1+\varepsilon) w(M_{\text{opt}}(V))$. We define a function f from $Z \to 2^E$ such that for each edge $(p_1, p_2) \in Z$, $f((p_1, p_2))$ forms a path between p_1 and p_2, $f((p_1, p_2)) \subseteq E$, and $w(f((p_1, p_2))) \le (1+\varepsilon) d(p_1, p_2)$.

(i) Let $p_1 \in b_1$, $p_2 \in b_2$, $b_1 \in B_i$, $b_2 \in B_i$, and $(p_1, p_2) \in (Z_i - Z_{i-1})$, $i \le \delta$. Then we define $f((p_1, p_2))$ to be the set that has smaller weight among the two sets of edges $\{(p_1, r(b_1)), (r(b_1), p_2)\}$ and $\{(p_2, r(b_2)), (r(b_2), p_1)\}$. From Lemma 3, $f((p_1, p_2)) \subseteq E_i$, and from Lemma 1, $w(f((p_1, p_2))) \le (1+\varepsilon) d(p_1, p_2)$.
(ii) If $(p_1, p_2) \in (Z - Z_\delta)$ then $(p_1, p_2) \in E$, and we let $f((p_1, p_2)) = \{(p_1, p_2)\}$.

Consider the graph $(V, \bigcup_{e \in M_{\text{opt}}(V)} f(e))$. This graph is a subgraph of $G = (V, E)$, and its weight is at most $(1+\varepsilon) w(M_{\text{opt}}(V))$. Each connected component of this graph must contain an even number of vertices which is seen as follows. Assume that there is a connected component C of $(V, \bigcup_{e \in M_{\text{opt}}(V)} f(e))$ which contains an odd number of vertices. Since $M_{\text{opt}}(V)$ is a complete matching on V there must be a vertex p in C which is matched to a vertex p' not in C by an edge in $M_{\text{opt}}(V)$. Then $f((p, p'))$ forms a path between $p \in C$ and $p' \notin C$ and this contradicts the assumption that C is a connected component of $(V, \bigcup_{e \in M_{\text{opt}}(V)} f(e))$. Hence from Lemma 4 below we may conclude that each connected component of $(V, \bigcup_{e \in M_{\text{opt}}(V)} f(e))$ contains a subgraph such that every vertex in the component has odd degree in the subgraph. Thus the graph $(V, \bigcup_{e \in M_{opt}(V)} f(e))$ contains a subgraph where every point in V has odd degree. This gives

$$w(\psi_{\text{opt}}(G)) \le w\left(\bigcup_{e \in M_{\text{opt}}(V)} f(e) \right) \le (1+\varepsilon) w(M_{\text{opt}}(V)).$$

The following lemma is proved in [9].

LEMMA 4. *Let T be a spanning tree on a set of vertices S and let $|S|$ be even. Then there is a subgraph T' of T such that every vertex in S has odd degree in T'.*

PROOF. A proof of Lemma 4 is given in [9] but we include it for completeness. Pair up the vertices in S in some way. (Each vertex is paired up with exactly one other vertex.) There is a unique path in T connecting two paired-up vertices. Let Π be the set of paths between paired-up vertices, and let T' be the set of those edges in T which are contained in an odd number of paths in Π. We show that an odd number of edges in T' are incident to any vertex in S. For an edge e in T, let $num(e)$ be the number of paths in Π which contain e. Let $v \in S$ and let $num(v) = \sum_{e \text{ incident to } v} num(e)$. The path in Π which terminates at v contributes one to $num(v)$, and any other path in Π contributes either zero or two to $num(v)$. Hence, $num(v)$ is odd, and there must be an edge e incident to v such that $num(e)$ is odd. Furthermore, the total number of edges $e \in T$ such that e is incident to v and $num(e)$ is odd must also be an odd number. Thus each vertex v in S has odd degree in T'. □

In addition, $\psi_{\text{opt}}(G)$ has the interesting property that each point in V has degree at most $c \log_2 n$ in $\psi_{\text{opt}}(G)$, for some constant c dependent on k. Using this property of G, it may be possible to find $\psi_{\text{opt}}(G)$ or an odd degree subgraph of G with weight close to $w(\psi_{\text{opt}}(G))$ in a manner more efficient than the one described in Section 6. Lemma 5 bounds the degree of each point in $\psi_{\text{opt}}(G)$.

LEMMA 5. *If $\varepsilon \le 1$ then each point p in V has degree at most $c \log_2 n$ in $\psi_{\text{opt}}(G)$ where $c = 8^{k+1}$.*

PROOF. Given in the Appendix. □

Next, we describe how to implement the algorithm for extracting G from V. To obtain a fast implementation of Algorithm *Sparse-Graph* we construct a data structure which is best described as a *tree-of-boxes*. The *tree-of-boxes* is quite similar to the cell-tree [2], [3] and the quadtree [10].

1. The root of the tree is the box g_0, and the children of each box b in B_i are those boxes in B_{i+1} which are subboxes of b.
2. The leaf boxes are the boxes in B_δ. At each leaf in the *tree-of-boxes* we store the point in V that is located in the leaf box.
3. The boxes at each level i, i.e., the boxes in B_i are linked together in a doubly linked list.
4. From each box b in B_i there are pointers to (i) its father in B_{i-1}, (ii) its sons in B_{i+1}, (iii) each box b' in B_i satisfying $d_{\min}(b, b') \le L_i$, and (iv) the leftmost leaf box in the subtree rooted at box b.

The *tree-of-boxes* has $O(\log n)$ levels and at most n boxes per level. For each box $b \in B_i$, there are at most $O((2c_q\varepsilon^{-1})^k)$ boxes b' in B_i such that $d_{\min}(b, b') \le L_i$. So the *tree-of-boxes* requires $O((2c_q\varepsilon^{-1})^k n \log n)$ storage, and can be constructed in $O((2c_q\varepsilon^{-1})^k n \log n)$ time by starting from the root and proceeding toward the leaves level by level.

Once the *tree-of-boxes* is available, the representatives in boxes may be chosen in time proportional to $|B| = O(n \log n)$. For boxes b_1 and b_2, we can find points $p_1 \in b_1$, $p_2 \in b_2$, such that $d(p_1, p_2) = d_{\min}(b_1, b_2)$ in $O(k)$ time. Then using the *tree-of-boxes*, each of the edge sets E_i can be extracted in $O(k(2c_q\varepsilon^{-1})^k n)$ time, and so G may be extracted in $O(k(2c_q\varepsilon^{-1})^k n \log n)$ time.

5. Reducing to a Problem with Bounded Ratio of Edge Lengths. In this section we describe how to partition the set V into $V_0, V_1, \ldots, V_m$, and in the process obtain a matching M_0 such that:

(i) M_0 is a complete matching on V_0 and $w(M_0) \leq w(M_{\text{opt}}(V))/n$.
(ii) For $1 \leq i < j \leq m$, $d_{\min}(V_i, V_j) > w(M_{\text{opt}}(\bigcup_{l=1}^{m} V_l))$.
(iii) For $1 \leq i \leq m$, $d_{\max}(V_i)/d_{\min}(V_i) \leq 1458k^2n^5 \leq n^8$, for $n \geq 12k^{2/3}$.

We make the following two observations. First, $M_{\text{opt}}(\bigcup_{i=1}^{m} V_i)$ cannot contain an edge joining a point in V_i to a point in V_j, for $i \neq j$. Thus

$$\sum_{i=1}^{m} w(M_{\text{opt}}(V_i)) = w\left(M_{\text{opt}}\left(\bigcup_{i=1}^{m} V_i\right)\right).$$

Second, the graph induced by the symmetric difference of the matchings M_0 and $M_{\text{opt}}(V)$ consists of alternating cycles and disjoint simple paths between pairs of vertices in $\bigcup_{i=1}^{m} V_i$. So by the triangle inequality,

$$w\left(M_{\text{opt}}\left(\bigcup_{i=1}^{m} V_i\right)\right) \leq w(M_0) + w(M_{\text{opt}}(V)).$$

Thus if M_i is a complete matching on V_i such that $w(M_i) \leq (1+\varepsilon)w(M_{\text{opt}}(V_i))$, for $1 \leq i \leq m$, then $\bigcup_{i=0}^{m} M_i$ is a complete matching on V such that $w(\bigcup_{i=0}^{m} M_i) \leq (1+\varepsilon+2(1+\varepsilon)/n)w(M_{\text{opt}}(V))$.

In Section 5.1 we describe how to partition V in $O(n \log n)$ time once we have an upper bound u on $w(M_{\text{opt}}(V))$ such that $u \leq 9nw(M_{\text{opt}}(V))$. In Section 5.2 we outline how such a bound u may be obtained in $O(n \log n)$ time.

5.1. Partitioning the Set of Points V. Let u be an upper bound on $w(M_{\text{opt}}(V))$ such that $u \leq 9nw(M_{\text{opt}}(V))$. We first split V into V_0 and $V - V_0$, and then split $V - V_0$ into $V_1, V_2, \ldots, V_m$. We briefly describe a data structure used in partitioning V. Given a set of points S and a parameter α, in $O(|S| \log|S|)$ time we can construct a data structure $D(S, \alpha)$ that has the following properties:

1. $D(S, \alpha)$ is a simple undirected graph.
2. With each vertex v in $D(S, \alpha)$ is associated a cubical box $b(v)$ of size α, and all the points in $b(v) \cap V$ are stored at vertex v.
3. For distinct vertices v and v', $b(v) \cap b(v') = \varphi$.
4. There is an edge between vertices v and v' iff $d_{\min}(b(v), b(v')) \leq \alpha$.
5. $\bigcup_{v \in D(S,\alpha)} (b(v) \cap V) = S$.

We briefly describe how to construct $D(S, \alpha)$ in $O(|S| \log|S|)$ time. The construction is an inductive construction. For $k = 1$ (the base case) $D(S, \alpha)$ is obtained as follows. The points are first sorted. Then the points are scanned in order, and while scanning the points are split into disjoint intervals of size α and links are created between pairs of intervals separated by a distance of at most α. Now assume that $k \geq 2$. Let $\bar{S}$ be the set of points obtained by projecting the points in S onto the plane $x_k=0$. (x_k denotes the k'th coordinate of a point in k-dimensions.) $D(S, \alpha)$ is constructed from $D(\bar{S}, \alpha)$ as follows. For a vertex $v \in D(\bar{S}, \alpha)$, let $\pi(v)$ be the set of those points p such that $p \in S$ and the projection of p onto the plane $x_k=0$ lies in the box $b(v)$. For each vertex $v \in D(\bar{S}, \alpha)$, we sort the points in $\pi(v)$ by x_k-coordinate, split the points in $\pi(v)$ into cubical boxes of size α by scanning them in increasing order of x_k-coordinate, and in the process construct a linked list of these boxes. The edges in $D(S, \alpha)$ are obtained by simultaneously scanning the lists of boxes corresponding to $\pi(v)$ and $\pi(v')$ for each edge (v, v') in $D(\bar{S}, \alpha)$.

We have the following useful lemma about the connected components of $D(S, \alpha)$. We note that the connected components of $D(S, \alpha)$ are almost identical to the neighbor-connected-components defined by Clarkson in [3].

LEMMA 6.

(I) *If $p \in b(v) \cap S$, $p' \in b(v') \cap S$, and v, v' are in distinct connected components of $D(S, \alpha)$, then $d(p, p') \geq \alpha$.*

(II) *Let C be a connected component of $D(S, \alpha)$. We can construct a spanning tree T on the points in $S \cap (\bigcup_{v \in C} b(v))$ such that each edge in T is of length at most $3k\alpha$.*

(III) *If $p \in b(v) \cap S$, $p' \in b(v') \cap S$, and v, v' are in the same connected component of $D(S, \alpha)$, then $d(p, p') \leq 3k\alpha|S|$.*

PROOF. (I) above follows from the definition of $D(S, \alpha)$. (III) follows from (II) and the application of the triangle inequality. A spanning tree T as required in (II) is obtained as follows. Let $\hat{T}$ be a spanning tree on the connected component C. For each edge $(v, v') \in \hat{T}$, we connect a point in $b(v) \cap S$ to a point in $b(v') \cap S$. Then for each vertex v in component C, we construct an arbitrary spanning tree on the points in $b(v) \cap S$. This gives a spanning tree T on the points in S that are stored at vertices in C. That each edge in T is of length at most $3k\alpha$ follows from the following two observations. First, if (v, v') is an edge in $D(S, \alpha)$ and $p \in b(v) \cap S$, $p' \in b(v') \cap S$, then $d(p, p') \leq 3k\alpha$ for any of the L_q metrics. Second, for a vertex v in $D(S, \alpha)$, $d_{\max}(b(v)) \leq k\alpha$. □

To partition V into V_0 and $V - V_0$ we construct $D(V, u/27kn^3)$ and find its connected components $C_1, \ldots, C_l$. Let q_i be the total number of points in V stored at vertices in component C_i. As described in Lemma 6 above, we can find a spanning tree T_i, of length at most $uq_i/9n^3$, on the q_i points in V that are stored in C_i. Once a spanning forest on $D(V, u/27kn^3)$ is available, T_i may be obtained in $O(q_i)$ time. In each such spanning tree T_i we choose some leaf point as a representative point, and let $V - V_0$ be the set of representative points in spanning trees containing an odd number of points. Next, delete all the points in $V - V_0$ from trees $T_1, \ldots, T_l$. Each of the spanning trees now contains an even number of points, and the collection of spanning trees may be converted into a complete matching M_0 on V_0, of weight at most $u/9n^2$, using the procedure described in Section 3. A tree is converted into a traveling salesman tour (with length at most twice the length of the tree) on the set of points in the tree, and the best of the two matchings induced by the tour is chosen.

To split $V - V_0$ into $V_1, \ldots, V_m$ we build $D(V - V_0, 2u)$. The sets of points stored in the m connected components of $D(V - V_0, 2u)$ are the sets $V_1, \ldots, V_m$, respectively. The entire process of splitting V, once u is available, takes $O(3^k n \log n)$ time.

We next show that the partition of V into $V_0, V_1, \ldots, V_m$, and the matching M_0 constructed above, have the properties (i), (ii), and (iii) mentioned at the beginning of Section 5. Since each edge in T_i is of length at most $u/9n^3$, $\sum_i w(T_i) \le u/9n^2$. Hence

$$\text{(i)} \qquad w(M_0) \le \sum_i w(T_i) \le \frac{u}{9n^2} \le \frac{w(M_{\text{opt}})}{n}.$$

Since distinct points in $V - V_0$ must be stored in distinct connected components of $D(V, u/27kn^3)$, by Lemma 6 we get

$$d_{\min}(V - V_0) \ge \frac{u}{27kn^3}.$$

As all points in V_i are in the same connected component of $D(V - V_0, 2u)$, by Lemma 6 we have,

$$\text{for } 1 \le i \le m, \qquad d_{\max}(V_i) \le 6kun.$$

Thus,

$$\text{(ii)} \qquad \text{for } 1 \le i \le m, \qquad \frac{d_{\max}(V_i)}{d_{\min}(V_i)} \le 1458k^2n^5.$$

It was shown in Section 5 that $w(M_{\text{opt}}(\bigcup_{i=1}^m V_i)) \le w(M_0) + w(M_{\text{opt}}(V))$. So from the bound on $w(M_0)$ it follows that

$$w\left(M_{\text{opt}}\left(\bigcup_{i=1}^{m} V_i\right)\right) < 2u.$$

Since points in V_i and points in V_j are stored in distinct connected components of $D(V - V_0, 2u)$ for $i \ne j$, from Lemma 6 we may also conclude that,

$$\text{(iii)} \qquad \text{for } 1 \le i < j \le m, \qquad d_{\min}(V_i, V_j) \ge 2u > w\left(M_{\text{opt}}\left(\bigcup_{l=1}^{m} V_l\right)\right).$$

5.2. Finding a Good Upper Bound on Length of Shortest Matching. We run a greedy heuristic to obtain a complete matching M on V such that $w(M) \le 9nw(M_{\text{opt}}(V))$, and let $u = w(M)$. The heuristic proceeds in stages. At the beginning of a stage we have a set of edges X, and a set of leftover vertices L, such that each vertex in $(V - L)$ has at least one edge in X incident on it and each connected component of the graph $(V - L, X)$ contains an even number of vertices. Initially, $V = L$ and $X = \varphi$.

During a stage we grow X and decrease the number of leftover vertices by at least a factor of three. For a vertex $p \in L$, let $nearest[p]$ denote a nearest neighbor of p in $L-\{p\}$, i.e., $nearest[p]$ is a point in $L-\{p\}$ such that $\forall p' \in (L-\{p\})$, $d(p, nearest[p]) \leq d(p, p')$. For each vertex p in L, we compute the nearest neighbors of p and let E_{nn} be the set of edges given by

$$E_{nn} = \{(p, nearest[p]): p \in L\}.$$

The remainder of the computation during a stage is as follows. Find the connected components of the graph (L, E_{nn}). Choose a representative vertex in each component such that upon removing the representative vertex the remaining vertices in the component still remain connected (note that such a vertex always exists). The new set of leftover vertices is the set of representatives in the components containing an odd number of vertices. All the edges in E_{nn} are added to X except those that are incident on a new leftover vertex.

In $\lceil \log_3 n \rceil$ stages the number of leftover vertices falls to zero. Let L_i and X_i be the sets L and X at the beginning of the ith stage. We show that $w(M_{\text{opt}}(L_{i+1})) \leq 3w(M_{\text{opt}}(L_i))$ and $w(X_{i+1}) \leq w(X_i) + 2w(M_{\text{opt}}(L_i))$. Let E^i_{nn} be the set E_{nn} computed during the ith stage. Since an edge in E^i_{nn} connects some point p in L_i to a nearest neighbor of p in $L_i - \{p\}$, we have

$$w(E^i_{nn}) \leq 2w(M_{\text{opt}}(L_i)),$$

and thus

$$w(X_{i+1}) \leq w(X_i) + w(E^i_{nn}) \leq w(X_i) + 2w(M_{\text{opt}}(L_i)).$$

Each connected component of the graph $(L_i - L_{i+1}, X_{i+1} - X_i)$ contains an even number of vertices, and so using the procedure in Section 3 we can construct a complete matching M_i on $L_i - L_{i+1}$ such that

$$w(M_i) \leq w(X_{i+1} - X_i) \leq w(E^i_{nn}) \leq 2w(M_{\text{opt}}(L_i)).$$

The symmetric difference of $M_{\text{opt}}(L_i)$ and M_i consists of disjoint alternating cycles and augmenting paths between pairs of vertices in L_{i+1}, and so by the triangle inequality there exists a complete matching on L_{i+1} whose weight is at most $w(M_{\text{opt}}(L_i)) + w(M_i)$. Thus

$$w(M_{\text{opt}}(L_{i+1})) \leq w(M_{\text{opt}}(L_i)) + w(M_i) \leq 3w(M_{\text{opt}}(L_i)).$$

Since $w(M_{\text{opt}}(L_{i+1})) \leq 3w(M_{\text{opt}}(L_i))$ and $w(X_{i+1}) \leq w(X_i) + 2w(M_{\text{opt}}(L_i))$, we conclude that when the above procedure terminates X satisfies the condition $w(X) \leq 9nw(M_{\text{opt}}(V))$, and X may be converted into a complete matching on V without increase in weight as described in Section 3. Using the All-Nearest-Neighbors algorithm in [13], at each stage E_{nn} may be found in $O(|L| \log |L|)$ time, and the entire procedure for computing the upper bound u may be implemented in $O(n \log n)$ time.

6. Odd Degree Subgraphs and Complete Matchings. In this section we show how to find an odd degree subgraph Γ of G, such that $w(\Gamma) \leq (1+1/n)w(\psi_{\text{opt}}(G))$, in $O((2c_q\varepsilon^{-1})^{1.5k}(\alpha(n, n))^{0.5}n^{1.5}(\log n)^{2.5})$ time. The problem of finding a suitable odd degree subgraph Γ of G is reduced to the problem of finding an almost minimum weight complete matching in a graph G' related to G. In Section 6.1 we describe how to construct a graph $G' = (V', E')$ from $G = (V, E)$ satisfying the following properties:

(a) $|V'| \leq 2|E| + |V|$ and $|E'| \leq 5(|E| + |V|)$.
(b) An odd degree subgraph Γ of G can be converted into a complete matching M' in G' such that $w(\Gamma) = w(M')$.
(c) A complete matching M' in G' can be converted into an odd degree subgraph Γ of G such that $w(\Gamma) \leq w(M')$.

From (b) and (c) above it follows that $w(\psi_{\text{opt}}(G)) = w(M_{\text{opt}}(G'))$. Thus to find a suitable odd degree subgraph Γ of G it suffices to find a complete matching M' in G' such that $w(M') \leq (1+1/n)w(M_{\text{opt}}(G'))$.

Such a matching M' in G' may be found using the scaling algorithm for weighted matching given in [6]. However, a faster way of finding such a matching M' is to utilize the new scaling algorithm in [7]. The following lemma is proved in [7].

WEIGHTED MATCHING LEMMA. *Given a weighted undirected graph $\hat{G} = (\hat{V}, \hat{E})$, a complete matching $\hat{M}$ in $\hat{G}$ such that $w(\hat{M}) \leq (1+1/N)w(M_{\text{opt}}(\hat{G}))$ can be found in $O((|\hat{V}|\alpha(|\hat{E}|, |\hat{V}|)\log(|\hat{V}|))^{0.5}|\hat{E}|\log(N|\hat{V}|))$ time, where α is the inverse Ackermann function* [12].

The time to obtain Γ is broken down as follows. Note that G' can be constructed from G in $O(|V| + |E|)$ time, and the conversion between an odd degree subgraph Γ of G and a complete matching M' of G' can also be done in $O(|V| + |E|)$ time. Thus once G is available, G' can be obtained in $O(|V| + |E|)$ time. Utilizing the scaling algorithm in [7] a complete matching M' in G', such that $w(M') \leq (1+1/n)w(M_{\text{opt}}(G'))$, can be found in

$$O((|V'|\alpha(|E'|, |V'|)\log(|V'|))^{0.5}|E'|\log(n|V'|))$$

time. M' can be converted into an odd degree subgraph Γ of G such that

$$w(\Gamma) \leq w(M') \leq (1+1/n)w(M_{\text{opt}}(G')) \leq (1+1/n)w(\psi_{\text{opt}}(G))$$

in $O(|V| + |E|)$ time. Finally, since $|E| = O((2c_q)^k\varepsilon^{-k}n\log n)$, $|V'| = O(|V| + |E|)$, $|E'| = O(|V| + |E|)$, and $\alpha(|E'|, |V'|) = O(\alpha(n, n))$, we get that the time to find a suitable odd degree subgraph Γ of G is $O((2c_q\varepsilon^{-1})^{1.5k}(\alpha(n, n))^{0.5}n^{1.5}(\log n)^{2.5})$.

6.1. Constructing G' from G. We show how to construct a graph $G' = (V', E')$ from the given graph $G = (V, E)$ satisfying the following properties:

(a) $|V'| \leq 2|E| + |V|$ and $|E'| \leq 5(|E| + |V|)$.
(b) An odd degree subgraph Γ of G can be converted into a complete matching M' in G' such that $w(\Gamma) = w(M')$.

(c) A complete matching M' in G' can be transformed into an odd degree subgraph Γ of G such that $w(\Gamma) \leq w(M')$.

Then an MWCM in G' and a minimum weight odd degree subgraph of G have the same weight.

Let $1, 2, \ldots, n$ denote the n vertices in V. We denote vertices in V' by ordered pairs of positive integers, an ordered pair with i on the first coordinate and l on the second coordinate will be represented by $[i, l]$ (note that an unordered pair of i and j is denoted by (i, j)). We let α_i denote the smallest odd number greater than or equal to the degree of vertex i in G. The edges incident on each vertex i in G will be assumed to have been ordered in some manner. For an ordered pair $[i, j]$ of vertices in V, we let $\pi([i, j]) = [l, m]$ if $(i, j) \in E$, (i, j) is the lth edge incident on i and the mth edge incident on j. G' is obtained from G as follows:

1. Corresponding to each vertex i in G, there is a cycle C_i of α_i vertices in G'. Let $[i, 0], \ldots, [i, \alpha_i - 1]$, denote the vertices in C_i, and let $s([i, l])$ denote $([i, (l+1) \bmod \alpha_i])$. If $\alpha_i > 1$, then in cycle C_i there is an edge of weight zero between $[i, l]$ and $s([i, l])$, for $1 \leq l \leq \alpha_i$.
2. Let $(i, j) \in E$ and let $\pi([i, j]) = [l, m]$. Corresponding to edge (i, j) in G there is a set of edges $F((i, j))$ in G', and each edge in $F((i, j))$ has the same weight as (i, j).

There are four cases for $F((i, j))$ depending on α_i and α_j.

Case 1. $\alpha_i = 1$, $\alpha_j = 1$,

$$F((i, j)) = \{([i, l], [j, m])\}.$$

Case 2. $\alpha_i = 1$, $\alpha_j > 1$,

$$F((i, j)) = \{([i, l], [j, m]), ([i, l], s([j, m]))\}.$$

Case 3. $\alpha_i > 1$, $\alpha_j = 1$,

$$F((i, j)) = \{([i, l], [j, m]), (s([i, l]), [j, m])\}.$$

Case 4. $\alpha_i > 1$, $\alpha_j > 1$,

$$F((i, j)) = \{([i, l], [j, m]), (s([i, l]), [j, m]), ([i, l], s([j, m])), (s([i, l]), s([j, m]))\}.$$

By construction, $|V'| \leq 2|E| + |V|$ and $|E'| \leq 5(|E| + |V|)$. The construction of G' from G is illustrated in Figures 1 and 2. Figure 1 gives a graph G with four vertices and four edges, and Figure 2 gives the corresponding graph G'.

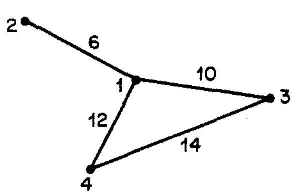

Fig. 1. Graph $G = (V, E)$. Ordering of edges: vertex 1, (1, 2), (1, 3), (1, 4); vertex 2, (2, 1); vertex 3, (3, 1), (3, 4); and vertex 4, (4, 1), (4, 3).

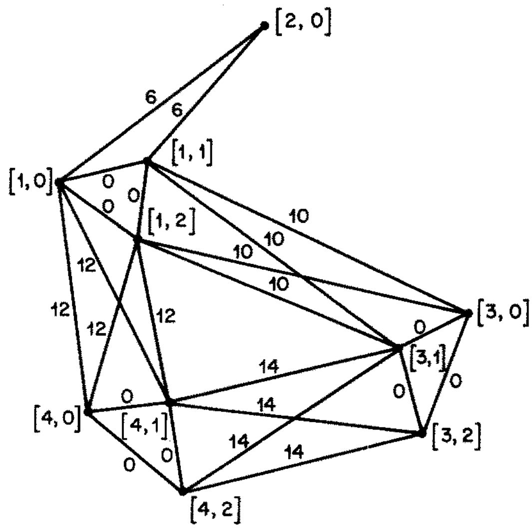

Fig. 2. Graph $G' = (V', E')$.

We first show that given an odd degree subgraph Γ of G we can obtain a complete matching M' in G' of the same weight. Let $\hat{M}$ be the matching given by

$$\hat{M} = \{([i, l], [j, m]) : (i, j) \in \Gamma, \pi([i, j]) = [l, m]\}.$$

The vertices in cycle C_i that are not matched by an edge in $\hat{M}$ are partitioned into intervals. Such an interval of unmatched vertices which contains an odd number of vertices is called an odd interval. Let the number of odd intervals in C_i be q_i. As $\hat{M}$ matches an odd number of vertices in C_i, q_i is even. We traverse cycle C_i as follows. We start with a matched vertex, and from vertex $[i, l]$ we move to vertex $s([i, l])$. During the traversal we number the odd intervals in C_i from 1 to q_i in the order they are encountered. While traversing C_i, we also mark each matched vertex in C_i that lies between two successive odd intervals numbered $2r-1$ and $2r$, $1 \le r \le (q_i/2)$. Thus the vertices in V' that are matched by an edge in $\hat{M}$ are divided into marked and unmarked vertices.

We shift the matching $\hat{M}$ so that the number of odd intervals in each C_i reduces to zero. If $q_i > 0$, then, for $1 \le r \le q_i/2$, shifting will add a vertex to the $(2r-1)$st odd interval in C_i, and remove a vertex from the $(2r)$th odd interval in C_i. Let

$$shift(([i, l], [j, m])) = \begin{cases} ([i, l], [j, m]), [i, l] \text{ unmarked}, [j, m] \text{ unmarked}, \\ (s([i, l]), [j, m]), [i, l] \text{ marked}, [j, m] \text{ unmarked}, \\ ([i, l], s([j, m])), [i, l] \text{ unmarked}, [j, m] \text{ marked}, \\ (s([i, l]), s([j, m])), [i, l] \text{ marked}, [j, m] \text{ marked}. \end{cases}$$

We obtain a matching M'' from $\hat{M}$ by replacing each edge e in $\hat{M}$ by $shift(e)$. As $w(shift(e)) = w(e)$, it follows that $w(M'') = w(\hat{M})$. The vertices in C_i that are not matched by an edge in M'' are divided into intervals, each containing an even number of vertices. Then M'' can be extended to a complete matching M' in G', by matching up the unmatched vertices by edges of weight zero.

The conversion of a complete matching M' in G' to an odd degree subgraph Γ of G goes as follows. For $i \neq j$, the number of edges in M' which join a vertex in C_i to a vertex in C_j can be either 1 or 2. Suppose the number is 2. Then these edges are $([i, l], [j, m])$ and $(s([i, l]), s([j, m]))$ for some l and m. We can then replace these two edges by the edges $([i, l], s([i, l]))$, $([j, m], s([j, m]))$, which have zero weight and thereby decrease the weight of the matching. By such replacements we can convert M' to a complete matching $\hat{M}$ in G' such that $w(\hat{M}) \leq w(M')$, and if $i \neq j$ there is at most one edge in $\hat{M}$ joining a vertex in C_i and a vertex in C_j. As each C_i has an odd number of vertices, the number of vertices in C_i that are matched to vertices outside C_i must be odd. Then we select an edge (i, j) to be in Γ iff there is an edge between C_i and C_j in $\hat{M}$. Γ is an odd degree subgraph of G and $w(\Gamma) = w(\hat{M}) \leq w(M')$.

Finally, we note that G' can be obtained from G in $O(|V|+|E|)$ time, and the conversion between an odd degree subgraph Γ of G and a complete matching M' in G' may be accomplished in $O(|V|+|E|)$ time.

7. Conclusion. Utilizing the underlying geometry, we have developed a fast algorithm that finds an almost minimum weight complete matching on a set of points in k-dimensional space. Given a set of n points, the algorithm runs in $O((2_{c_q})^{1.5k}\varepsilon^{-1.5k}(\alpha(n, n))^{0.5}n^{1.5}(\log n)^{2.5})$ time, and the weight of the complete matching obtained is at most $(1+\varepsilon)$ times the weight of an MWCM on the given set of points. Here $c_q = 6k^{1/q}$ for the L_q-metric, α is the inverse Ackermann function, and ε is a parameter less than or equal to 1.

Acknowledgments. The author would like to thank Professor C. L. Liu, S. Kapoor, and P. Ramanan for helpful discussions.

Appendix. In this appendix we give a proof of Lemma 5 in Section 4.

LEMMA 5. *If $\varepsilon \leq 1$ then each point p in V has degree at most $c \log_2 n$ in $\psi_{\text{opt}}(G)$ where $c = 8^{k+1}$.*

PROOF. Let $E_L \subseteq E$ such that the ratio of the maximum to the minimum edge lengths in E_L is at most 2. We show that there are at most 8^k edges in $\psi_{\text{opt}}(G) \cap E_L$ incident on any point p, and then since the ratio of the maximum to the minimum edge lengths in G is at most n^8, it follows that any point p in V has degree at most $8^{k+1} \log_2 n$ in $\psi_{\text{opt}}(G)$. Assume there are m edges $(p, p_1), \ldots, (p, p_m)$ in $\psi_{\text{opt}}(G) \cap E_L$ incident on a point p in V. Let $d_{\min}(E_L)$ be the length of a shortest edge in E_L. Among the points $p_1, \ldots, p_m$ there cannot be a pair p', p'', such that $d(p', p'') < d_{\min}(E_L)$. Let us assume that there do exist such points p', p''. In

Section 4 we showed the existence of a function f from $Z \to 2^E$ such that for each edge $(p_1, p_2) \in Z$, $f((p_1, p_2))$ forms a path between (p_1, p_2), $f((p_1, p_2)) \subseteq E$, and $w(f((p_1, p_2))) \leq (1+\varepsilon)d(p_1, p_2)$. Thus for $\varepsilon \leq 1$, there is a path P in G between p' and p'', of length at most $2d(p', p'') < 2d_{\min}(E_L)$. Then the graph induced by the set of edges $\psi_{\text{opt}}(G) - \{(p, p'), (p, p'')\} \cup P$ has smaller weight than $\psi_{\text{opt}}(G)$, and by Lemma 4 this graph also contains a subgraph where every point in V has odd degree. This would contradict the minimality of $\psi_{\text{opt}}(G)$. All the points $p_1, \ldots, p_m$ are located in a ball C_p of radius $2d_{\min}(E_L)$ centered at p. For $1 \leq i \leq m$, let C_{p_i} be the ball of radius $d_{\min}(E_L)/2$ centered at p_i. No two of the balls $C_{p_1}, \ldots, C_{p_m}$ intersect, and the intersection of C_{p_i} and C_p contains a ball of radius $d_{\min}(E_L)/4$. Then since the number of disjoint balls of radius $d_{\min}(E_L)/4$ that can be packed in a ball of radius $2d_{\min}(E_L)$ is at most 8^k, m cannot exceed 8^k. □

References

[1] D. Avis, A survey of heuristics for the weighted matching problem, *Networks*, **13**, 1983, 475–493.

[2] K. L. Clarkson, Fast algorithms for the all nearest neighbors problem, *Proc. IEEE Symp. on Foundations of Computer Science*, 1983, pp. 226–232.

[3] K. L. Clarkson, Fast expected-time and approximation algorithms for geometric minimum spanning trees, *Proc. 16th Annual ACM Symp. on Theory of Computing*, 1984, pp. 342–348.

[4] J. Edmonds, Matching and a polyhedron with 0–1 vertices, *J. Res. Nat. Bur. Standards*, **69B**, 1965, 125–130.

[5] H. Gabow, An efficient implementation of Edmond's algorithm for maximum matching on graphs, *J. Assoc. Comput. Mach.*, **23**, 1976, 221–234.

[6] H. Gabow, A scaling algorithm for weighted matching on general graphs, *Proc. 26th Annual Symp. on Foundations of Computer Science*, 1985, pp. 90–100.

[7] H. N. Gabow and R. E. Tarjan, Faster algorithms for graph matching, Technical Report, Department of Computer Science, Princeton University, Princeton, NJ.

[8] M. Iri, M. Murota, and S. Matsui, Linear time heuristics for minimum-weight perfect matching on a plane with an application to the plotter algorithm. Unpublished (1980).

[9] D. A. Plaisted, Heuristic matching for graphs satisfying the triangle inequality, *J. Algorithms*, **5**, 1984, 163–179.

[10] H. Samet, The quadtree and related hierarchical data structures, Technical Report, Department of Computer Science, University of Maryland, College Park, MD.

[11] K. J. Supowit and E. M. Reingold, Divide-and-conquer heuristics for minimum weighted Euclidean matching, *SIAM J. Comput.*, **12**, 1983, 118–143.

[12] R. E. Tarjan, *Data Structures and Network Algorithms*, SIAM, Philadelphia, PA, 1983.

[13] P. M. Vaidya, An optimal algorithm for the all-nearest-neighbors problem, *Proc. 27th Annual Symp. on Foundations of Computer Science*, 1986.