A Faster Algorithm for Computing the Link Distance Between Two Point Sets on the Real Line

Justin Colannino Godfried Toussaint
School of Computer Science
McGill University
Montréal, Québec, Canada

Technical Report SOCS-2005.5, September 2005.

Abstract

Let S and T be point sets with $|S| \geq |T|$ and total cardinality n. A *linking* between S and T is a matching, L, between the sets where every element of S and T is matched to *at least* one element of the other set. The *link* distance is defined as the minimum-cost linking. In this note we consider a special case of the link distance where both point sets lie on the real line and the cost of matching two points is the distance between them in the L_1 metric. An $O(n^2)$ algorithm for this problem is presented, improving the previous best known complexity of $O(n^3)$.

1 Introduction

Let S and T be point sets with $|S| \geq |T|$ and total cardinality n. A *linking* between S and T is a matching, L, between the sets where all elements of S and T are matched to *at least* one element of the other set. The *link* distance between the two sets is defined as the minimum-cost linking according to some distance function.

This distance measure was originally proposed in 1997 by Eiter and Mannila [5] in the context of measuring the relationship between theories expressed in a logical language. Eiter and Mannila show that this problem can be solved in $O(n^3)$ time via a reduction to the computation of a minimum-weight perfect matching in a suitable bipartite graph. Their algorithm is discussed in further detail in section 2.

The link distance can also be expressed as the *minimum-weight bipartite edge cover* problem. For a weighted bipartite graph $G = (S \cup T, w, E)$, the edge cover problem finds a subset, E', of edges such that each vertex is the endpoint of at least one edge. The minimum-weight edge cover problem finds the set E' of minimum weight. This problem may be solved in $O(n^3)$ time using the Hungarian method [8].

The link distance has also been solved in the more general setting of *bibranchings*, first introduced by Schrijver [9]. Let $D = (V,E)$ be a directed graph, and let V be partitioned into two disjoint sets, the *source* vertices S and the *target* vertices T. A *bibranching* in D with respect to S is a set of edges B in E such that:

for each v in S, B contains a directed path from v to a vertex in T, and

for each v in T, B contains a directed path from a vertex in S to v.

For the special case when D is a bipartite graph with color classes S and T, and all the edges in D are directed from S to T, the bibranching is a bipartite edge cover.

Keijsper and Pendavingh [7] describe an $O(|E|)$ time algorithm attributed to J. F. Geelen for reducing the minimum-weight bipartite edge cover problem to the maximum-weight matching problem. They also describe a solution for the latter problem that uses shortest path algorithms from [4] and [10] sped up with Fibonacci heaps [6]. Their algorithm runs in time $O(n'(|E| + n\log n))$, where $n' = min\{|S|,|T|\}$. In our problem this complexity is $O(n^3)$ in the worst case, matching the complexity of the approach of Eiter and Mannila. However, the algorithm of Eiter and Mannila is simpler.

In this note we address the special case where both point sets lie on the real line and the distance is measured with the L_1 metric. This version of the problem has applications to sequence comparison in bioinformatics [2] as well as measuring music similarity in music information retrieval [11]. We provide a $O(n^2)$ algorithm improving the previous best complexity of $O(n^3)$.

2 The $O(n^3)$ Algorithm of Eiter and Mannila

Since our $O(n^2)$ algorithm is inspired by, and based on, some results of Eiter and Mannila [5], we describe in this section their main results for the sake of clarity and completion. Eiter and Mannila's solution to finding a minimum-cost linking reduces the problem to finding a minimum-weight matching in a bipartite graph. The reduction depends on the following two lemmas, originally proved by Eiter and Mannila in [5].

Lemma 2.1. *Let S and T be two sets of points, and L a minimum-cost linking between them. For $s \neq s'$ if $(s,t) \in L$, and $(s',t) \in L$ then the distance from s to t is not more than the distance from s to any other element of T.*

Proof. Assume that for some minimum-cost linking L the above property does not hold. This implies that there is some s where $(s,t) \in L$ such that $\delta(s,t) > \delta(s,t')$ for some $t' \in T$, and there is $s' \neq s$ where $(s',t) \in L$. Consider L^* which differs from L only in that s is linked to t' instead of t. Then the cost of L^* is less than the cost of L, which contradicts the assumption that L is a minimum-cost linking. □

Lemma 2.2. *Let S and T be two sets of points and L be a minimum-cost linking between them. Then, for any relation $(s,t) \in L$, either s or t has degree 1.*

Proof. Assume that there exists a minimum-cost linking L for which the lemma does not hold. This implies that for some s,t where $(s,t) \in L$ there exist s',t' where the relations (s,t') and (s',t) are also in L. But then we can construct a new linking L^* which differs from L only in the exclusion of (s,t). L' is also a linking, since all elements are linked to at least one element of the other set; yet the cost of L^* is less than cost of L, a contradiction. □

Eiter and Mannila proceed by constructing a bipartite graph in which the matching is performed. The key concept is the creation of dummy nodes to handle the case when an element, x, of either set is linked to an element of the other set which has degree more than one, and hence, by Lemma 2.1 is x's nearest neighbor in that set. From lemma 2.2 we know that any such x must have degree one. Therefore we need only create one copy for each element of $S \cup T$ with the weight of an edge between an element and its dummy node equal to the distance between that element and its nearest neighbor.

The desired bipartite graph consists of two complete subgraphs; one representing the elements of S and T with weights equal to the distances between the elements; the other a complete zero-weight dummy subgraph. An element in the dummy subgraph is connected to its corresponding element in the first subgraph with a weight equal to the distance to that element's nearest neighbor.

More formally, we construct such a bipartite graph G'' as follows. For sets S and T create a graph $G = (A \cup B, E, w)$ in the following manner. For each $s \in S$ create an $a \in A$, and for each $t \in T$ create a $b \in B$. For each edge, $e = (a_i, b_j)$, $w(e) = \delta(s_i, t_j)$. Next, create a zero weight copy of the graph $G' = (A' \cup B', E', w')$. From these two graphs we construct a graph G'' which is the union of G and G' and contains additional edges from a_i to a'_i and b_j to b'_j with weight equal to the minimum distance from s_i to an element of T and from t_j to an element of S, respectively.

It remains to be shown that the weight of the minimum-weight matching, $w(M)$, is equal to the cost of the minimum-cost linking $c(L)$. We proceed by first showing $w(M) \leq c(L)$, and then $w(M) \geq c(L)$.

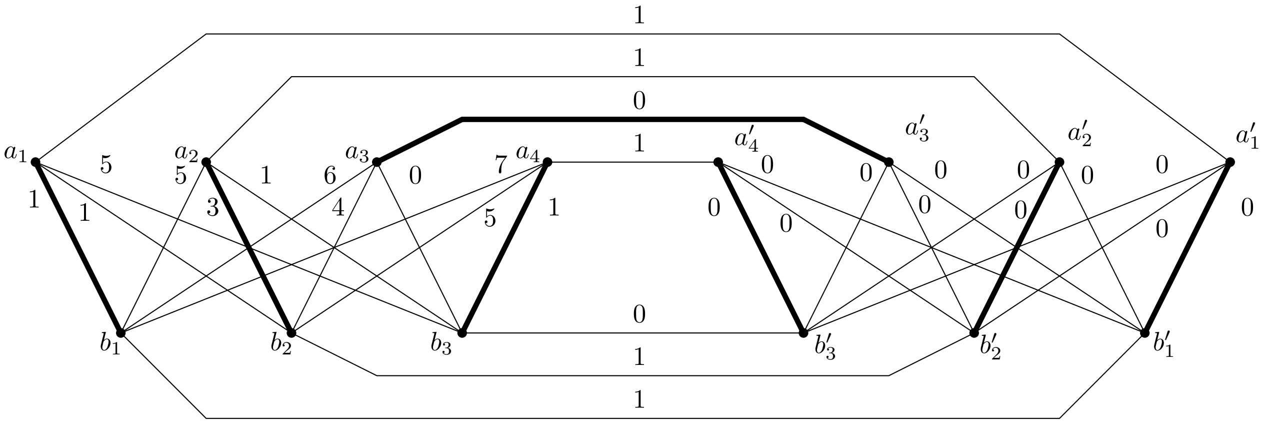

Figure 1: A graph corresponding to S = {2,6,7,8}, T = {1,3,7}. The minimum-weight perfect matching is given in bold.

Lemma 2.3. $w(M) \leq c(L)$

Proof. From a minimum-cost linking L we construct a matching M^* as follows. For every pairing $(s_i,t_j) \in L$ we perform one of three actions. If both s_i and t_j have not yet been matched, then we add the edges (a_i,b_j) and (a'_i,b'_j) to M. Otherwise, if s_i is already matched, we add (b_j,b'_j) to the matching. Finally, if t_j is already matched, we add (a_i,a'_i) to the matching. Note that by lemma 2.2 both s_i and t_j cannot already have been matched. Since each element s_i or t_j is linked in L the corresponding nodes in G'' must be matched. Therefore M is a matching. Furthermore, the weight of M^* is equal to the cost of L because for the cost of each individual linking (s_i,t_j) there is a corresponding matching in M^* of equal weight. This follows from the definition of the weight function and lemma 2.1. Thus $w(M^*) = c(L)$ establishing that $w(M) \leq c(L)$. □

Lemma 2.4. $w(M) \geq c(L)$

Proof. From a minimum-weight matching M we construct a linking L^*. For every matching $m \in M$ if $m = (a_i,b_j)$ we add (s_i,t_j) to L. Otherwise, if $m = (a_i,a'_i)$ then we add a link between s_i and $t \in T$ where t minimizes $\delta(s_i,t)$. Likewise for $m = (b_j,b'_j)$ we add a link between b_j and an $s \in S$ that minimizes $\delta(s,b_j)$. This must be a linking since each element in S and T is represented by some a_i or b_j which must occur in some matching. The weight of M is less than the cost of L because the cost of each link created is equal to the corresponding match by virtue of the manner in which the weight function is defined. We conclude that $c(L) \leq w(M)$. □

Theorem 2.5. $w(M) = c(L)$

Proof. The result is immediate from lemmas 2.3 and 2.4. □

Letting n equal to $|S|$, Eiter and Mannila note that this reduction yields an $O(n^3)$ time algorithm since the construction of the bipartite graph takes $O(n^2)$ time, and finding a perfect matching in a complete bipartite graph takes $O(n^3)$ time [3].

3 The Proposed $O(n^2)$ Algorithm

For two sets of points on the real line we can improve upon the previous $O(n^3)$ algorithm by using the following simple lemma, sometimes called the quadrangle inequality [1].

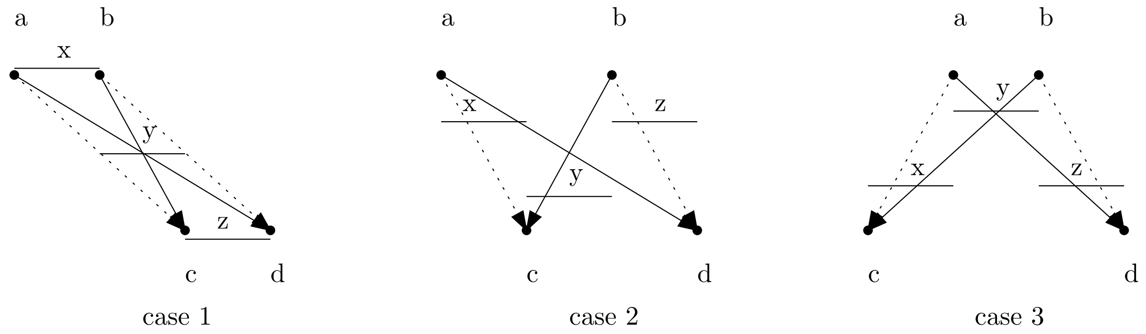

Figure 2: Illustration of lemma 3.1. The dotted lines represent the smaller distance.

Lemma 3.1. *Let S and T be sets of points on the line. Also, let a distance function $\delta(s,t) = |s-t|$. Then for $a,b \in S$ where $a < b$, and $c,d \in T$ where $c < d$,*

$$\delta(a,c) + \delta(b,d) \leq \delta(a,d) + \delta(b,c) \tag{1}$$

Proof. Three cases arise as pictured in Figure 2.
Case 1 : *a and b are both less than c or, symmetrically, greater than d*

Let $x = |a-b|, y = |b-c|$ and $z = |c-d|$. Then we have $\delta(a,c) + \delta(b,d) = x + 2y + z = \delta(a,d) + \delta(b,c)$.

Case 2 : *Either a or b, but not both, are in between c and d*

Let $x = |a-c|, y = |c-b|$ and $z = |b-d|$. Then, $\delta(a,c) + \delta(b,d) = x + z \leq x + 2y + z = \delta(a,d) + \delta(b,c)$.

Case 3 : *Both a and b are in between c and d*

Let $x = |c-a|, y = |a-b|$ and $z = |b-d|$. Then, $\delta(a,c) + \delta(b,d) = x + z \leq x + 2y + z = \delta(a,d) + \delta(b,c)$

□

This lemma leads to the following corollary concerned with the link distance.

Corollary 3.2. *Let S and T be sets of points on the real line. Then there exists a minimum-cost linking L^* from S to T such that for all $s_i < s_j$, $L^*(s_i) \leq L^*(s_j)$.*

This observation implies that in a minimum-cost linking, if we know that $(s_i, t_j) \in L$, then one of (s_i, t_{j+1}), (s_{i+1}, t_j), or (s_{i+1}, t_{j+1}) must also be in L. Using this information it is not hard to reduce the problem to finding the shortest path through a weighted directed acyclic graph. The construction of this graph differs from the previous directed acyclic graph in that it is expanded to allow multiple linkings not only from elements of S to elements of T, but also from elements of T to elements of S.

Let S and T be sets of integers on the interval $(0, X)$. Let s_i denote the i^{th} element of S. We construct a directed acyclic graph G in the following way. For all pairs of elements s_i, t_j where $s_i \in S$ and $t_j \in T$ construct a vertex $v_{i,j}$. From each vertex $v_{i,j}$ add an edge to $v_{i,j+1}, v_{i+1,j}, v_{i+1,j+1}$ with weights $|t_i - s_{j+1}|, |t_{i+1} - s_j|, |t_{i+1} - s_{j+1}|$, respectively, provided that each vertex exists. Finally create a vertex labeled 'start' and insert an edge to $v_{1,1}$ with weight $|s_1 - t_1|$.

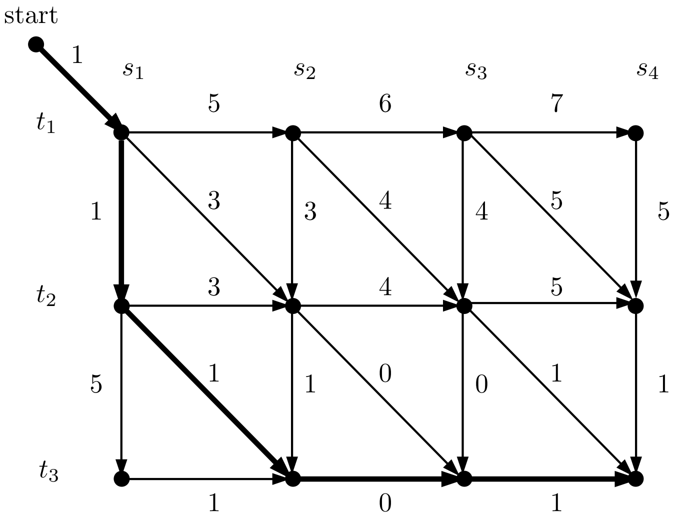

Figure 3: A directed acyclic graph corresponding to the sets S = {2, 6, 7, 8}, T = {1, 3, 7}. The bold edges indicate the minimum-weight path through the graph.

It remains to be shown that this construction makes the cost $c(L)$ of a minimum-cost linking on S and T, equal to the weight $w(P)$ of the shortest path between 'start' and $v_{|T|,|S|}$. To do so we proceed as before by first showing that $w(P) \leq c(L)$, and then showing that $w(P) \geq c(L)$.

Lemma 3.3. $w(P) \leq c(L)$

Proof. Consider a minimum-cost linking L on S and T. Then we can create a path P' from 'start' to $v_{|T|,|S|}$ in G in the following way. First, the edge from 'start' to $v_{1,1}$ is inserted. Next, for each (t_i, s_j) in L we know that one and only one of $(t_i, s_{j+1}), (t_{i+1}, s_j), (t_{i+1}, s_{j+1})$ is also in L. Therefore we add an edge from $v_{i,j}$ to whichever of the three that is in P'. Since such an edge must exist in the graph, the path is connected. Note that there is a one-to-one correspondence between edges and links, and by construction, each edge has the same weight as it's corresponding link. Therefore we conclude that $w(P') = c(L)$ and therefore, $w(P) \leq c(L)$. □

Lemma 3.4. $w(P) \geq c(L)$

Proof. From a minimum-weight path P from 'start' to $v_{|S|,|T|}$ in G, we create a linking, L', on the corresponding strings S and T. For any vertex $v_{i,j}$ through which P passes, add (s_i, t_j) to L. This is a valid linking because there exist no paths in the graph that do not touch all rows and columns, and thus each node in $S \cup T$ is included in some linking in L'. Since the weight of each edge used is equal to the cost of the corresponding linking we conclude that $w(P) = c(L')$, which yields $w(P) \geq c(L)$, the desired result. □

Theorem 3.5. $w(P) = c(L)$

Proof. The result is immediate from lemmas 3.3 and 3.4. □

As for the complexity of this method, let $|S| = n$ and $|T| = m$. In the construction $|V| = O(n * m)$, and since there are at most three edges per node, $|E| = O(|V|)$. Thus the construction takes $O(|V|) = O(n * m)$ time. The algorithm for finding the shortest path between two elements in a directed acyclic graph takes time $O(|V| + |E|)$ which in this case is $O(|V|) = O(n * m)$ time [3] . Thus the total time complexity of the method is $O(n * m) = O(n^2)$.

References

[1] Alok Aggarwal, Amotz Bar-Noy, Samir Khuller, Dina Kravets, and Baruch Schieber. Efficient minimum cost matching and transportation using the quadrangle inequality. *Journal of Algorithms*, 19(1):116–143, 1995.

[2] J. Colannino and G. Toussaint. An algorithm for computing the restriction scaffold assignment problem in computational biology. *Information Processing Letters*, 95(Issue 4):466–471, 2005.

[3] Thomas C. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press (second edition), Cambridge Mass., 2001.

[4] J. Edmonds and R. M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the Association for Computing Machinery,* 19:248–264, 1972.

[5] Thomas Eiter and Heikki Mannila. Distance measures for point sets and their computation. *Acta Informatica*, 34(2):109–133, 1997.

[6] M. L. Fredman and R. E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the Association for Computing Machinery*, 34:596–615, 1987.

[7] J. Keijsper and R. Pendavingh. An efficient algorithm for minimum-weight bibranching. *Journal of Combinatorial Theory*, 73(Series B):130–145, 1998.

[8] H.W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 2:83–97, 1955.

[9] A. Schrijver. Min-max relations for directed graphs. *Annals of Discrete Mathematics*, 16:261–280, 1982.

[10] N. Tomizawa. On some techniques useful for the solution of transportation network problems. *Networks*, 1:173–194, 1972.

[11] Godfried Toussaint. A comparison of rhythmic similarity measures. In *Proc. 5th International Conference on Music Information Retrieval,* pages 242–245, 2004.