Efficient Many-To-Many Point Matching in One Dimension

Justin Colannino[1], Mirela Damian[2], Ferran Hurtado[3], Stefan Langerman[4], Henk Meijer[5], Suneeta Ramaswami[6], Diane Souvaine[7], Godfried Toussaint[8]

[1] School of Computer Science, McGill University, Montreal, Quebec, Canada. Supported by NSERC (Canada). e-mail: `jcolan@cs.mcgill.ca`
[2] Department of Computer Science, Villanova University, Villanova, USA. e-mail: `mirela.damian@villanova.edu`
[3] Departament de Matemàtica Aplicada II, Universitat Politècnica de Catalunya, Barcelona, Spain. Partially supported by projects MCYT BFM2003-00368, MEC MTM2006-01267 and Gen. Cat. 2005SGR00692. e-mail: `Ferran.Hurtado@upc.edu`
[4] Chercheur qualifié du FNRS, Département d' Informatique, Université Libre de Bruxelles, Brussels, Belgium. e-mail: `stefan.langerman@ulb.ac.be`
[5] School of Computing, Queen's University, Kingston, Canada. e-mail: `henk@cs.queensu.ca`
[6] Department of Computer Science, Rutgers University, Camden, New Jersey 08102, USA. e-mail: `rsuneeta@camden.rutgers.edu`
[7] Department of Computer Science, Tufts University, Medford, USA. Partially supported by the National Science Foundation, Grant No. CCF-0431027. e-mail: `dls@cs.tufts.edu`
[8] School of Computer Science and Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT), The Schulich School of Music, McGill University, Montreal, Canada. Supported by FQRNT and NSERC. e-mail: `godfried@cs.mcgill.ca`

Appears in *Graphs and Combinatorics*, vol. 23 (2007), supplement, Computational Geometry and Graph Theory. The Akiyama-Chvatal Festschrift. The original publication is available at www.springerlink.com.

Abstract. Let S and T be two sets of points with total cardinality n. The minimum-cost *many-to-many* matching problem matches each point in S to at least one point in T and each point in T to at least one point in S, such that sum of the matching costs is minimized. Here we examine the special case where both S and T lie on the line and the cost of matching $s \in S$ to $t \in T$ is equal to the distance between s and t. In this context, we provide an algorithm that determines a minimum-cost many-to-many matching in $O(n \log n)$ time, improving the previous best time complexity of $O(n^2)$ for the same problem.

1. Introduction

Consider two finite sets of points S and T with total cardinality n. The problem of establishing a correspondence between the points in S and the points in T arises in various applications in computational biology [1], operations research [2], pattern recognition [3], computer vision [8], music information retrieval [20] and computational music theory [21]. One method of defining and measuring such a relationship uses the concept of *matching*. A matching between two sets is a function that pairs individual points in one set with individual points in the other. A *one-to-one* matching between S and T is a perfect matching between the two sets [13]. A *many-to-one* matching maps each element of S to *exactly* one element of T and each element of T to *at least* one element of S [6]. A many-to-many matching between two sets maps each element of S to *at least* one element of T and vice-versa [2]. The quality of a matching is measured by a cost function δ that assigns a cost $\delta(s,t)$ to each matched pair (s,t). The cost of a matching is the sum of the costs of all matched pairs (s,t), with $s \in S$ and $t \in T$.

Our result. In this paper we discuss the special case where the sets S and T (not necessarily disjoint) lie on the real line, and the cost $\delta(s,t)$ is defined as the distance between s and t. In this setting, we present an $O(n \log n)$ time algorithm for the minimum-cost many-to-many matching problem, and note that this is optimal: $\Omega(n \log n)$ is a lower bound for the time complexity of such an algorithm on unsorted sets S and T, by reduction from set equality. If the point sets S and T are given in sorted order, our matching algorithm runs in optimal $O(n)$ time, and this complexity matches the bound for the many-to-one and one-to-one matching problems for the same special case [4,13].

Background. The problem of many-to-many matching has been first studied by Eiter and Mannila [11] in the context of *link distance*, as a measure of similarity between two theories expressed in a logical language, and represented by point sets in a metric space.

In a graph theoretic setting, the many-to-many matching problem can be reduced to the *minimum-weight bipartite edge cover* problem. For a complete bipartite graph $G = (S \cup T, w, E)$, the minimum-weight edge cover problem seeks to find a subset of E of minimum-weight, such that every vertex in $S \cup T$ is adjacent to at least one edge.

The many-to-many matching problem has also been implicitly considered in the more general setting of *bibranchings* first introduced by Schrijver [17]. Let $D = (V, E)$ be a directed graph, and let V be partitioned into two disjoint sets, a set S of *source* vertices and a set T of *target* vertices. A *bibranching* in D with respect to S is a set of edges $B \subseteq E$ 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.

For arbitrary weighted graphs, the many-to-many matching problem has an $O(n^3)$-time solution. Indeed, Eiter and Mannila [11] achieve this bound via reduction to the minimum-weight perfect matching problem in a bipartite graph, which can be solved in $O(n^3)$ time using the Hungarian method. Keijsper and Pendavingh [14] 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 [9] and [19], sped up with Fibonacci heaps [12]. Their algorithm runs in time $O(n'(|E| + n \log n))$, where $n' = min\{|S|, |T|\}$; this time complexity is $O(n^3)$ in the worst case, thus matching the complexity of the simpler approach of Eiter and Mannila [11]. For the one dimensional case it was previously shown in [5], and in more detail in [7], that the many-to-many matching problem has an $O(n^2)$ solution via reduction to the problem of finding the shortest path through a directed acyclic graph.

The new $O(n \log n)$ time algorithm proposed here is described in section 3, before which some properties of an optimum many-to-many matching for point sets on the line are presented in section 2.

2. Properties of an Optimal Many-to-Many Matching

This section is concerned with the nature of pairings allowed in an optimal matching. Let S and T be two sets of points on the real line, and assume without loss of generality that the point with the smallest x-coordinate lies in S. For ease of presentation, we use the same symbol a to refer to both the point a and its x-coordinate in the plane; therefore, an expression such as $a < b$ (read a *smaller* than b) represents the fact that the x-coordinate of a is smaller than the x-coordinate of b. Furthermore, for ease of visualization, in the figures, we separate the points of S and T vertically.

We begin with defining a partition of $S \cup T$ into subsets $A_0, A_1, A_2, \ldots$ such that all points in A_i are smaller than all points in A_{i+1} for all i, A_0 is a maximal subset of consecutive points in S, A_1 is a maximal subset of consecutive points in T, A_2 is a maximal subset of consecutive points in S, and so forth (see ahead Figure 2).

Lemma 1. *If $b \in T$ and $c \in S$ are such that $b < c$, then a minimum-cost many-to-many matching contains no pairs (a, d) with $a \in S$, $d \in T$ and $a < b < c < d$.*

Proof. Suppose that the lemma is false. Let $\mathcal{M}$ be a minimum-cost many-to-many matching that contains such a pair (a, d). Replace (a, d) in $\mathcal{M}$ by the two pairs (a, b) and (c, d): the result $\mathcal{M}'$ is still a many-to-many matching. Furthermore, $\mathcal{M}'$ has a smaller cost than $\mathcal{M}$, since $(d-a) = (d-c)+(c-b)+(b-a) > (d-c)+(b-a)$ (see Figure 1a). This contradicts our assumption that $\mathcal{M}$ is a minimum-cost many-to-many matching. □

Corollary 1. *Any matching (a, d) in a minimum-cost many-to-many matching, with $a < d$, satisfies $a \in A_i$ and $d \in A_{i+1}$, for some $i \geq 0$.*

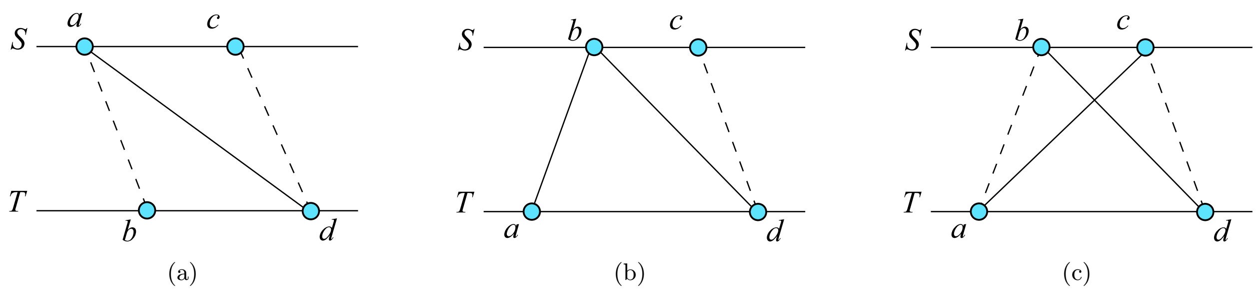

Fig. 1. Suboptimal matchings. (a) (a, d) is a suboptimal matching. (b) (a, b) and (b, d) do not both belong to an optimal matching. (c) (a, c) and (b, d) do not both belong to an optimal matching.

Lemma 2. *Let $b < c$ be two points in S. If a and d are two points in T such that $a \leq b < c \leq d$, then a minimum-cost many-to-many matching does not contain both of (a, b) and (b, d).*

Proof. Suppose that the lemma is false. Let $\mathcal{M}$ be a minimum-cost many-to-many matching that contains both (a, b) and (b, d) (see Figure 1b). Remove the pair (b, d) from $\mathcal{M}$ and add (c, d): the result $\mathcal{M}'$ is still a many-to-many matching. Furthermore, since $(d - b) > (d - c)$, $\mathcal{M}'$ has a smaller cost, a contradiction. □

Lemma 3. *Let $b < c$ be two points in S, and a and d two points in T such that $a \leq b < c \leq d$. Then a minimum-cost many-to-many matching does not contain both of (a, c) and (b, d).*

Proof. Suppose that the lemma is false. Let $\mathcal{M}$ be a minimum-cost many-to-many matching that contains both (a, c) and (b, d) (see Figure 1c). Replace (a, c) and (b, d) in $\mathcal{M}$ by the two other pairs (a, b) and (c, d): the result $\mathcal{M}'$ is still a many-to-many matching. Furthermore, since $(b - a) + (d - c) > (d - b) + (c - a)$, $\mathcal{M}'$ has a smaller cost, a contradiction. □

Lemma 4. *For each $i > 0$, A_i contains a point q_i such that, in a minimum-cost many-to-many matching, all points in A_i less than q_i are matched to points in A_{i-1} and all points in A_i greater than q_i are matched to points in A_{i+1}.*

Proof. If A_i contains a single point, the lemma is clearly true. We now discuss the case $|A_i| > 1$. Assume for contradiction that the lemma is false. First note that, if a point $a \in A_i$ is paired with a point $b < a$, then b must be in A_{i-1} (cf. Corollary 1). Similarly, if a is paired with $b > a$, then $b \in A_{i+1}$. Thus, if the lemma does not hold, there exist $a \in A_{i-1}$, $b, c \in A_i$ and $d \in A_{i+1}$ such that $a < b < c < d$ and both (a, c) and (b, d) are contained in a minimum-cost many-to-many matching. But this contradicts Lemma 3. □

Lemma 4 constitutes the basis of our dynamic programming approach discussed in section 3.

3. Matching Algorithm

Our dynamic programming matching algorithm seeks to determine the points q_i defined in Lemma 4 quickly. Once these points are determined, a minimum-cost matching can be easily computed, as described in Theorem 1.

For any point q, let $C(q)$ denote the cost of a minimum-cost many-to-many matching for the set of points $\{p \in S \cup T, \text{ with } p \leq q\}$.

Theorem 1. *Let S, T be sets of sorted points on the line. Then a minimum-cost many-to-many matching between S and T can be determined in linear time.*

Proof. We compute $C(p_i)$ for all points p_i in $S \cup T$; the computation of a matching of cost $C(p_i)$ is implicit from the computation of $C(p_i)$. If m is the largest point in $S \cup T$, then $C(m)$ is the minimum cost of a many-to-many matching.

For all points $p \in A_0$, we define $C(p) = \infty$. Assume that we have computed $C(p)$ for all points p in $A_0, \ldots, A_w$, for some $w \geq 0$. In the following we show how to compute $C(p)$ for all points $p \in A_{w+1}$ in $O(|A_w| + |A_{w+1}|)$ time, which implies the theorem. First we settle some notation and definitions.

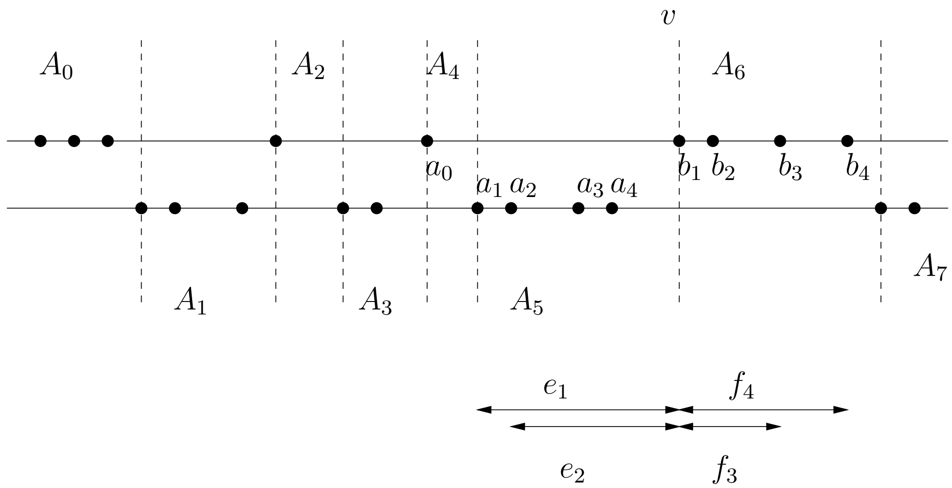

Fig. 2. Partition of point set $S \cup T$; notation and definitions.

Let $s = |A_w|$ and $t = |A_{w+1}|$. Let $A_w = \{a_1, a_2, \ldots, a_s\}$ with $a_1 < a_2 < \ldots < a_s$. Let $A_{w+1} = \{b_1, b_2, \ldots, b_t\}$ with $b_1 < b_2 < \ldots < b_t$. When $w > 0$, let a_0 denote the point of A_{w-1} of largest x-coordinate. Let v be the vertical line though b_1. Let e_i denote the horizontal distance between a_i and v. Let f_i denote the horizontal distance between v and b_i. These definitions are illustrated in Figure 2 for $w = 5$. Note that $f_1 = 0$. Recall that our goal is to compute $C(b_i)$, for each $b_i \in A_{w+1}$. We discuss five cases, depending on the values of w, s and t.

Case 0: $w = 0$**.** Assume first that $i \leq s$. In this case, a minimum cost is obtained by assigning the first $s - i$ elements of A_0 to b_1 and the remaining i elements pairwise, as depicted in Figure 3a. We compute the cost $C(b_i)$, for all $1 \leq i \leq \min(s, t)$:

$$C(b_i) \;=\; \sum_{j=1}^{s} e_j + \sum_{j=1}^{i} f_i.$$

Assume now that $i > s$. In this case, $C(b_i)$ is minimized when the first s points in A_1 are matched pairwise with the points in A_0 and the remaining $(i - s)$ points in A_1 are matched to a_s, as depicted in Figure 3b. So the value $C(b_i)$, for $\min(s, t) < i \leq t$, is:

$$C(b_i) \;=\; (i - s)e_s + \sum_{j=1}^{s} e_j + \sum_{j=1}^{i} f_i.$$

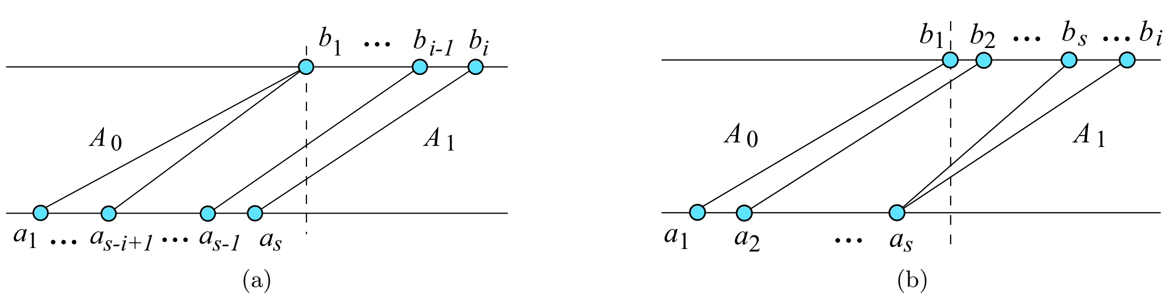

Fig. 3. Case 0: $w = 0$. (a) $1 \leq i \leq s$. (b) $s < i \leq t$.

Case 1: $w > 0,\ s = t = 1$**.** Lemma 4 implies that b_1 must be paired with a_1 (see Figure 4a). Consequently, the pair (a_1, a_0) accounted for in computing $C(a_1)$ should not be accounted for in computing $C(b_1)$, unless used to cover a_0. We identify two cases: (i) a_1 is paired with both b_1 and a_0 (and possibly other points in A_{w-1}), and (ii) a_1 is paired with only b_1. In the first case, $C(b_1)$ includes $C(a_1)$; in the second case, $C(b_1)$ includes $C(a_0)$. We choose the matching of minimum cost:

$$C(b_1) \ = \ e_1 + \min \ (C(a_0), C(a_1)).$$

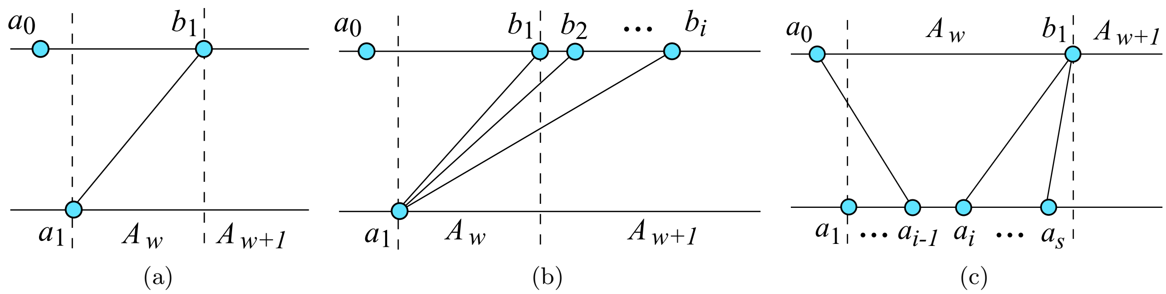

Fig. 4. (a) Case 1: $w > 0,\ s = t = 1$. (b) Case 2: $w > 0,\ s = 1,\ t > 1$. (c) Case 3: $w > 0,\ s > 1,\ t = 1$.

Case 2: $w > 0,\ s = 1,\ t > 1$**.** This case is similar to Case 1, the only difference being that *all* points in A_{w+1} are assigned to a_1, as depicted in Figure 4b. As before, a_1 may be assigned to other points in A_{w-1}, in which case $C(b_1)$ includes $C(a_1)$; otherwise, $C(b_1)$ includes $C(a_0)$. Therefore, for all $1 \leq i \leq t$, we compute:

$$C(b_i) \ = \ \sum_{j=1}^{i} f_j + ie_1 + \min \ (C(a_0), C(a_1)).$$

Case 3: $w > 0,\ s > 1,\ t = 1$**.** According to Lemma 4, we need to find the point q in A_w such that all points less then q are matched to points in A_{w-1} and all points greater than q are matched to points in A_{w+1}. Refer to Figure 4c. This is the point a_i that minimizes the quantity on the right hand side of the equation:

$$C(b_1) \ = \ \min_{i=1}^{s} \ (\sum_{j=i}^{s} e_j + C(a_{i-1})).$$

A matching of cost $C(b_1)$ would include all pairs (a_j, b_1), for all $j \geq i$, along with all pairs corresponding to $C(a_{i-1})$, as depicted in Figure 4c.

Case 4: $w > 0,\ s > 1,\ t > 1$**.** Let $S_i = \sum_{j=i}^{s} e_j + C(a_{i-1})$ for $i = 1, 2, \ldots, s$. Here S_i represents the cost of connecting points $a_i, a_{i+1}, \ldots, a_s$ to line v, plus the cost $C(a_{i-1})$. Let $M_i = \min\{S_j \mid 1 \leq j \leq i\}$. In other words, for a fixed i, M_i represents the smallest of $S_1, S_2, \ldots, S_i$. Again, we are looking for a point q in A_w that splits the matching to the left and right. To this end, for $1 \leq i \leq \min(s, t)$ we now compute three values:

$$X(b_i) \ = \ M_{s-i} + \sum_{j=1}^{i} f_j, \quad 1 \leq i < s,$$

$$Y(b_i) \ = \ \sum_{j=s-i+1}^{s} e_j + \sum_{j=1}^{i} f_j + C(a_{i-1}), \quad 1 \leq i \leq s,$$

and

$$Z(b_i) \ = \ \min_{j=s-i+2}^{s} \ (\sum_{h=j}^{s} e_h + \sum_{j=1}^{i} f_j + (i + j - s - 1)e_s + C(a_{j-1})), \quad 1 < i.$$

The quantities X, Y and Z above represent the following costs: $X(b_i)$ represents the cost of connecting $b_1, b_2, \ldots, b_i$ to at least $i+1$ points in A_w, as depicted in Figure 5a; $Y(b_i)$ represents the cost of connecting $b_1, b_2, \ldots, b_i$ to exactly i points in A_w, as depicted in Figure 5b; and $Z(b_i)$ represents the cost of connecting

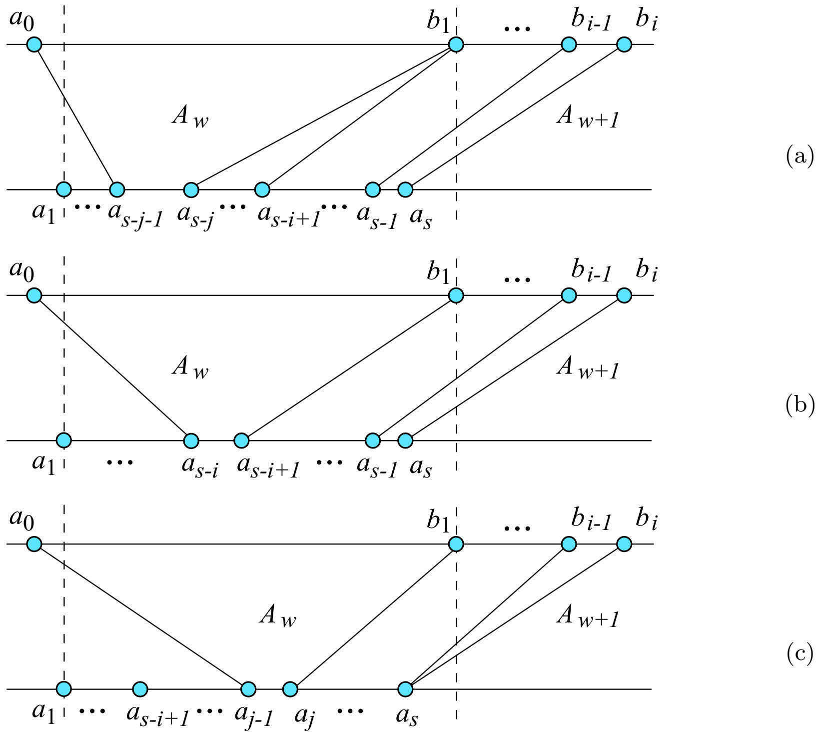

Fig. 5. Case 4: $w > 0,\ s > 1,\ t > 1$. (a) Computing $X(b_i)$. (b) Computing $Y(b_i)$. (c) Computing $Z(b_i)$.

$b_1, b_2, \ldots, b_i$ to fewer than i points in A_w, as depicted in Figure 5c. So $C(b_i)$ is the minimum of $X(b_i)$, $Y(b_i)$ and $Z(b_i)$.

It is not hard to see that the values $X(b_i)$ and $Y(b_i)$ can be computed in $O(s + t)$ time. Also note that

$$Z(b_i) \ = \ e_s + f_i + \min(Y(b_{i-1}), Z(b_{i-1})),$$

and therefore we can also compute $Z(b_i)$ for all $1 \le i \le \min(s, t)$, in $O(s+t)$ time. Finally, for $\min(s, t) < i \le t$ we have

$$C(b_i) \ = \ C(b_{i-1}) + e_s + f_i,$$

and so we can compute $C(b_i)$ for all $1 \le i \le t$, in $O(s + t)$ time. □

Figure 6 shows the minimum-cost many-to-many matching produced by this algorithm on 20 points.

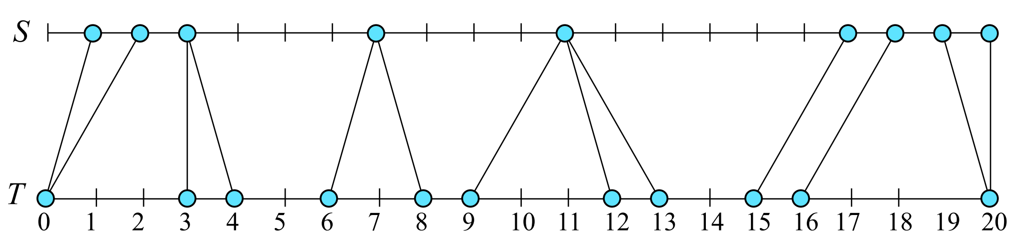

Fig. 6. Minimum-cost matching for a complete example: $|S| = 8$, $|T| = 12$, minimum many-to-many matching cost is 16.

4. Concluding Remarks

The *many-to-many* matching problem considered here was motivated by the one-dimensional problem concerned with musical rhythm in which the dimension is time [20], [21]. In the more general setting of melody matching, however, the problem may be viewed as two-dimensional, where the x-axis measures time, and the y-axis measures pitch. Thus the onsets of the notes in a melody may be represented as a point set in two dimensions. Empirical studies in music perception have shown that the L_1 metric works well in this context for measuring the distance between two points in the time-pitch plane [18], [15]. Generalizing our work to this two-dimensional version of the problem remains open. It is expected that since the complexity of the classic matching problems may be reduced by exploiting geometric information [22], [16], [10], a similar behavior will be observed with the *many-to-many* problem in two dimensions.

5. Acknowledgements

This research was started at the *20th Winter Workshop on Computational Geometry* organized by Godfried Toussaint at the Bellairs Research Institute of McGill University in Holetown, St. James, Barbados, West Indies, held from January 28 to February 4, 2005. The theme of the workshop was *Computational Geometry for Music Information Retrieval*. The authors thank the other participants for creating a stimulating environment and lively discussions: Greg Aloupis, David Bremner, Erik Demaine, Vida Dujmovic, Jeff Erickson, Francisco Gomez, John Iacono, Danny Krizanc, Erin McLeish, Pat Morin, Mark Overmars, David Rappaport, Ileana Streinu, Perouz Taslakian, Remco Veltcamp, Sue Whitesides, and David Wood. Many thanks to a referee for insightful comments.

References

1. A. Ben-Dor, Richard M. Karp, B. Schwikowski, and R. Shamir. The restriction scaffold problem. *Journal of Computational Biology*, 10(2):385–398, 2003.
2. R. E. Burkard and E. Çela. Linear assignment problems and extensions. In D.-Z. Du and P.M. Pardalos, editors, *Handbook of Combinatorial Optimization – Supplement Volume A*, volume 4, pages 75–149, Dordrecht, 1999. Kluwer Academic Publishers.
3. Samuel R. Buss and Peter N. Yianilos. A bipartite matching approach to approximate string comparison and search. Technical report, NEC Research Institute, Princeton, New Jersey, 1995.
4. J. Colannino, M. Damian, F. Hurtado, J. Iacono, H. Meijer, S. Ramaswami, and G. Toussaint. A $O(n \log n)$-time algorithm for the restricted scaffold assignment problem. *Journal of Computational Biology*, 13(4), 2006.
5. J. Colannino and G. Toussaint. Faster algorithms for computing distances between one-dimensional point sets. In Francisco Santos and David Orden, editors, *Proceedings of the XI Encuentros de Geometria Computacional*, pages 189–198, Santander, Spain, June 27-29 2005.
6. Justin Colannino and Godfried Toussaint. An algorithm for computing the restriction scaffold assignment problem in computational biology. *Information Processing Letters*, 95(4):466–471, August 2005.
7. Justin Colannino and Godfried Toussaint. A faster algorithm for computing the link distance between two point sets on the real line. Technical Report SOCS-TR-2005.5, McGill University, School of Computer Science, July 2005.
8. M. Fatih Demirci, Ali Shokoufandeh, Yakov Keselman, Lars Bretzner, and Sven Dickinson. Object recognition as many-to-many feature matching. *International Journal of Computer Vision*, 69(2):203–222, 2006.
9. 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.
10. Alon Efrat, A. Itai, and M. J. Katz. Geometry helps in bottleneck matching and related problems. *Algorithmica.*, 31:1–28, 2001.
11. Thomas Eiter and Heikki Mannila. Distance measures for point sets and their computation. *Acta Informatica*, 34(2):109–133, 1997.
12. 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.
13. Richard M. Karp and S.-Y.R. Li. Two special cases of the assignment problem. *Discrete Mathematics*, 13(46):129–142, 1975.
14. J. Keijsper and R. Pendavingh. An efficient algorithm for minimum-weight bibranching. *Journal of Combinatorial Theory*, 73(Series B):130–145, 1998.

15. Michael Keith. *From Polychords to Pólya: Adventures in Musical Combinatorics.* Vinculum Press, Princeton, 1991.
16. O. Marcotte and S. Suri. Fast matching algorithms for points on a polygon. *SIAM J. Comput.*, 20:405–422, 1991.
17. A. Schrijver. Min-max relations for directed graphs. *Annals of Discrete Mathematics*, 16:261–280, 1982.
18. James Tenney and Larry Polansky. Temporal gestalt perception in music. *Journal of Music Theory*, 24(2):205–241, Autumn 1980.
19. N. Tomizawa. On some techniques useful for the solution of transportation network problems. *Networks*, 1:173–194, 1972.
20. Godfried T. Toussaint. A comparison of rhythmic similarity measures. In *Proceedings of the 5th International Conference on Music Information Retrieval*, pages 242–245, Barcelona, Spain, October 10-14 2004. Universitat Pompeu Fabra.
21. Godfried T. Toussaint. The geometry of musical rhythm. In *Proceedings of the Japan Conference on Discrete and Computational Geometry*, volume LNCS 3742, pages 198–212, Berlin-Heidelberg, 2005. Springer-Verlag.
22. P. M. Vaidya. Geometry helps in matching. *SIAM J. Comput.*, 18:1201–1225, 1989.