Available online at http://scik.org
J. Math. Comput. Sci. 5 (2015), No. 2, 160-171
ISSN: 1927-5307

WEIGHTED RECTILINEAR MIN-SUM DISTANCE PROBLEM AND ITS GENERALIZATION

LAURENCE BOXER

Department of Computer and Information Sciences, Niagara University, New York 14109, USA; and Department of Computer Science and Engineering, State University of New York at Buffalo

Abstract. The Weighted Rectilinear Min-Sum Distance Problem is a problem in optimization: given a finite set of points $S = \{p_i\}_{i=0}^{n-1} \subset R^d$ that might represent client locations, and a set of positive numbers $\{w_i\}_{i=0}^{n-1}$ that might represent the relative importances of efficient deliveries to the respective client locations p_i, find a point $p_* \in R^d$ such that the total of the weighted rectilinear distances, $\sum_{i=0}^{n-1} w_i l_1^d(p_i, p_*)$, is minimized, where l_1^d is the l_1 or "city block" metric in the Euclidean d-dimensional space R^d. A flawed sequential solution to this problem is given in [10]. In this paper, we discuss improvements that can be made in the presentation of [10], and we give a parallel implementation of our algorithm. We also present a solution to a generalization of the Weighted Rectilinear Min-Sum Distance Problem.

Keywords: weighted rectilinear min-sum distance problem, parallel computer, PRAM

2010 AMS Subject Classification: 68U05, 68Q25, 68W10.

1. Introduction

E-mail address: boxer@niagara.edu

Received December 2, 2014

In this paper, we study the *Weighted Rectilinear Min-Sum Distance Problem (WRMSDP)*, considered in [10], and a generalization of this problem. The WRMSDP is described in [10] as follows.

> ... we are given a finite set ... $S = \{p_0, p_1, \ldots, p_{n-1}\}$ of n points in R^d with positive weights $w_0, w_1, \ldots, w_{n-1}$, and the goal is to find a point $c_* = (c_{*,0}, c_{*,1}, \ldots, c_{*,d-1}) \in R^d$ that minimizes the sum of the weighted rectilinear distances (with l_1 norm) between the points in S and c_*.
>
> A possible application of this problem is finding a location for a service center in a town in which all the streets are parallel to the axes. The location of the service center should minimize the sum of the weighted rectilinear distance between n customer locations given in S, and the service center.... we may want to give the distance to certain sites more "weight" in the calculation, for example if a site requires several deliveries daily as opposed to one delivery daily to other sites.... Or, perhaps we want to ensure particularly good service to a certain site.

The word "clients" need not suggest individual persons. E.g., a slightly different interpretation is that the clients could be stores of a retail chain that wishes to serve its outlets efficiently by building a new distribution center.

We also consider the more general (k,m) *Weighted Rectilinear Min-Sum Distance Problem*. In this problem, we are given sets $S = \{p_i\}_{i=0}^{n-1} \subset R^d$, $C = \{c_j\}_{j=0}^{k-1} \subset R^d$, a set of positive weights $\{w_i\}_{i=0}^{n-1}$, and a positive integer m. We assume $k \geq 0$, $m \geq 1$, and $k+m < n$; in practice, we often have $k+m \ll n$. The goal is to find a set of points $C_* = \{c_{*,j}\}_{j=0}^{m-1} \subset R^d$ such that the sum of the weighted rectilinear distances from members of S to nearest members of $C \cup C_*$ is minimized. Note that if $k = 0$ (i.e., if $C = \emptyset$) and $m = 1$, then this problem reduces to the WRMSDP.

The (k,m) WRMSDP corresponds to a corporation or service organization that has k existing service centers and wishes to build m additional service centers. As above, members of S could be clients' locations. Members of C could be locations of existing service centers. The points of C_* could then be the locations of new service centers.

These problems belong to a body of research concerning facility location in operations research. Related papers include [1,3,4,5,6,8,9].

The l_1 or "city block" distance is described as follows. Given points $x = (x_1, x_2, \dots, x_d)$, $y = (y_1, y_2, \dots, y_d) \in R^d$, the l_1 metric in dimension d is defined by

$$l_1^d(x,y) = \sum_{i=1}^{d} |x_i - y_i|.$$

Notice that

$$l_1^d(x,y) = \sum_{i=1}^{d} l_1^1(x_i, y_i). \tag{1}$$

From equation (1), Zhu and Wang [10] observe that, in order to obtain a solution for the WRMSDP in R^d, one can solve the problem for every coordinate separately. A similar observation holds for the (k,m) WRMSDP.

We also note that Zhu and Wang assume the input in R is sorted. This assumption cannot be maintained in higher dimensions; we will find it useful to have the data ordered with respect to individual coordinates, but data ordered with respect to one coordinate is not generally ordered with respect to other coordinates. Therefore, our version of the algorithm has a sort step. As a result, this algorithm for the WRMSDP in R takes $\Theta(n \log n)$ sequential time, rather than the $\Theta(n)$ time asserted by Zhu and Wang under the assumption that the input is sorted.

We give both sequential and parallel solutions for the (k,m) WRMSDP.

2. Preliminaries

2.1 (k,m) WRMSDP

Given the set $S = \{p_i\}_{i=0}^{n-1} \subset R^d$, the set $C = \{c_j\}_{j=0}^{k-1} \subset R^d$, and a positive integer m, our goal is to compute a set $C_* = \{c_{*,u}\}_{u=0}^{m-1} \subset R^d$ that produces a minimal value for the expression

$$\sum_{i=0}^{n-1} w_i \min\{l_1^d(p_i, y) \,|\, y \in C \cup C_*\}. \tag{2}$$

Equivalently, given the set $S = \{p_i\}_{i=0}^{n-1} \subset R^d$, the set $C = \{c_j\}_{j=0}^{k-1} \subset R^d$, and a positive integer m, our goal is to compute a set $C_* = \{c_u\}_{u=0}^{m-1} \subset R^d$ that produces a maximal decrease in the weighted sum of distances between "clients" and "service centers" when the set of service centers is expanded from C to $C \cup C_*$. I.e., we wish to maximize

$$\sum_{i=0}^{n-1} w_i \min\{l_1^d(p_i, y) \mid y \in C\} - \sum_{i=0}^{n-1} w_i \min\{l_1^d(p_i, y) \mid y \in C \cup C_*\}. \tag{3}$$

2.2 Fundamental operations

A *semigroup operation* uses a binary associative operation $\oplus$ that is applied to n data values $x_0, x_1, \ldots, x_{n-1}$ to obtain the cumulative result $x_1 \oplus \ldots \oplus x_{n-1}$. If a single binary computation of the $\oplus$ operator performs in $\Theta(1)$ time, then a semigroup operation can be performed sequentially in $\Theta(n)$ time [7].

A *parallel prefix operation* uses a binary associative operation $\oplus$ that is applied to n data $x_0, x_1, \ldots, x_{n-1}$ to obtain not only the result of a semigroup operation, but also a set of partial results, namely, all of

$$x_0,\ x_0 \oplus x_1,\ \ldots,\ x_0 \oplus x_1 \oplus \ldots \oplus x_{n-1}.$$

If a single binary computation of the $\oplus$ operator performs in $\Theta(1)$ time, then a parallel prefix operation can be performed sequentially in $\Theta(n)$ time [7].

Examples of binary operators commonly used for semigroup and parallel prefix operations include *sum, product, max, min, AND, OR*.

2.3 PRAM

The *parallel random access machine (PRAM)* (see [7]) is a model of parallel computation in which identical processors $P_0, P_1, \ldots, P_{n-1}$ share access to a global memory. In this paper, we assume a PRAM has the *exclusive write (EW)* property, meaning that only one processor can write to a given memory location during a clock cycle. The exclusive read (ER) property allows only one processor to read a memory location during a clock cycle. The concurrent read (CR) property permits multiple processors to read a given memory location during a clock cycle. Thus, given the same data and the same number of processors, if an EREW PRAM (a PRAM with the ER and EW properties) solves a given problem of size n in $T(n)$ time, then a CREW PRAM (a PRAM with the CR and EW properties) can solve the same problem in $O(T(n))$ time.

An EREW PRAM of n processors can perform the following operations on n data in the running times given.

Semigroup operation	$\Theta(\log n)$ [7]
Parallel prefix	$\Theta(\log n)$ [7]

We use the following.

Theorem 2.1. [2] An EREW PRAM of n processors can sort n data in $\Theta(\log n)$ time. $\square$

3 Solution of WRMSDP

In this section, we show how to solve the WRMSDP. The algorithm we present uses ideas developed in [10], but is somewhat simpler than the sequential algorithm of the latter paper; also, our algorithm lends itself more easily to parallelization.

We recall observations in [10] concerning the 1-dimensional version of the WRMSDP.

Theorem 3.1 [10] The WRMSDP can be solved in R in $\Theta(n)$ sequential time if the input is sorted. $\square$

Zhu and Wang noted that the function that measures the total of the weighted distances from members of S to a point $x \in R$,

$$f(x) = \sum_{i=0}^{n-1} w_i |p_i - x|,$$

is piecewise linear. Suppose $S = \{p_i\}_{i=0}^{n-1}$ is sorted so that $p_0 \le p_1 \le \ldots \le p_{n-1}$. Let $p_{-1} = -\infty$, $p_n = \infty$. Then, observed Zhu and Wang,

$$\text{for } x \in [p_{j-1}, p_j], \text{ we have } f(x) = f_j(x) = k_j x + b_j \tag{4}$$

where

$$k_0 = -\sum_{i=0}^{n-1} w_i,\ k_{j+1} = k_j + 2w_j,\ 0 \le j \le n-1 \tag{5}$$

and

$$b_0 = \sum_{i=0}^{n-1} w_i x_i,\ b_{j+1} = b_j - 2w_j x_j,\ 0 \le j \le n-1. \tag{6}$$

Also, from the piecewise linearity of the function $f(x)$, we have the following.

Theorem 3.2 [10] For the WRMSDP in R, there is a solution c_* for the location of the service center such that $c_* \in P$. □

Careless reading might lead one to conclude incorrectly that Zhu and Wang claim $\max\{f(x) \,|\, x \in R\}$ is finite even under the assumption that all the weights w_i are positive (see equation (4) of [10]). A careful reader will note that it follows easily from the assumption that the weights w_i are positive that $\max\{f(x) \,|\, x \in R\} = \infty$.

Theorem 3.3 A solution to the WRMSDP in R can be computed in $\Theta(n \log n)$ sequential time and in $\Theta(\log n)$ time on an EREW PRAM of n processors. I.e., given $S = \{p_i\}_{i=0}^{n-1} \subset R$ and $\{w_i\}_{i=0}^{n-1}$ such that $w_i > 0$ for all i, a point $x^* \in R$ such that $f(x^*) = \min\{f(x) \,|\, x \in R\}$ can be computed in $\Theta(n \log n)$ sequential time and in $\Theta(\log n)$ time on an EREW PRAM of n processors.

Proof: Consider the following algorithm.

(1) Sort S into ascending order. This is done in $\Theta(n \log n)$ sequential time. By Theorem 2.1, this is done in $\Theta(\log n)$ time on the PRAM.

(2) Compute k_0 according to the first of equations (5), using a semigroup operation. This takes $\Theta(n)$ sequential time, or $\Theta(\log n)$ time on the PRAM.

(3) Compute $\{k_i\}_{i=1}^{n-1}$ according to the second of equations (5), using a parallel prefix operation. This is done in $\Theta(n)$ sequential time and in $\Theta(\log n)$ time on a PRAM.

(4) Compute b_0 according to the first of equations (6), using a semigroup operation. This takes $\Theta(n)$ sequential time, or $\Theta(\log n)$ time on the PRAM.

(5) Compute $\{b_i\}_{i=1}^{n-1}$ according to the second of equations (6), using a parallel prefix operation. This is done in $\Theta(n)$ sequential time and in $\Theta(\log n)$ time on a PRAM.

(6) For $0 \leq i \leq n-1$, do

Let $f(x)$ be as in equation (4).

If $k_i \geq 0$ then

$$\min\{f(x) \,|\, x \in [x_i, x_{i+1}]\} = f_i(x_i) = k_i x_i + b_i,$$

so let $x_i' = x_i$ and $m_i = f_i(x_i)$; otherwise,

$$\min\{f(x) \,|\, x \in [x_i, x_{i+1}]\} = f_i(x_{i+1}) = k_i x_{i+1} + b_i,$$

so let $x'_i = x_{i+1}$ and $m_i = f_i(x_{i+1})$.

End For. This step executes in $\Theta(n)$ sequential time and in $\Theta(1)$ time on the PRAM.

(7) Use a semigroup computation to obtain a value $x^* \in \{x'_i\}_{i=0}^{n-1} \subset S$ such that

$$f(x^*) = \min\{m_i\}_{i=0}^{n} = \min\{f(x) \,|\, x \in R\}.$$

This requires $\Theta(n)$ sequential time or $\Theta(\log n)$ time on the PRAM.

Clearly, the algorithm runs in $\Theta(n \log n)$ sequential time and in $\Theta(\log n)$ time on the PRAM. $\square$

Corollary 3.4 The WRMSDP in R^d can be solved in $\Theta(dn \log n)$ sequential time, or in $\Theta(d \log n)$ time on a PRAM.

Proof: As noted in [10] and discussed above, a solution can be found by applying the solution in R to each coordinate. Thus, we have the following algorithm.

For $j = 1$ to d, do

Apply the algorithm of Theorem 3.3 to the set $S_j = \{p_{i,j}\}_{i=0}^{n-1}$ of the j^{th} coordinates of the members of S to obtain x_j^* such that

$$\sum_{i=0}^{n-1} w_i |p_{i,j} - x_j^*| = \min\{\sum_{i=0}^{n-1} w_i |p_{i,j} - x| \mid x \in R\}.$$

End For

The desired point is $x^* = (x_1^*, x_2^*, \ldots, x_d^*)$. The assertion follows immediately from Theorem 3.3. $\square$

4 Solution of (k,m) WRMSDP

In this section, we give a solution for the (k,m) WRMSDP in R.

Lemma 4.1 Any $c_* \in R$ that is a member of a solution C_* to a given instance of the (k,m) WRMSDP must satisfy

$$c_* \in [\min\{p_i \in S\}, \max\{p_i \in S\}].$$

Proof: If $c_* < \min\{p_i \in S\}$, then for all j,

$$w_j |p_j - c_*| > w_j |p_j - \min\{p_i \in S\}|.$$

FIGURE 1. Clients x_i and service centers c_j. Locating a new service center in $(-\infty, c_0]$ could change the service center nearest to x_0, but not to any $x_i > c_0$. Locating a new service center c_* in $[c_0, c_1]$ could change the service center nearest to x_1, but not to any x_i outside $[c_0, c_1]$. Locating a new service center c_* in the interval $[c_1, c_2]$ will not change the nearest center to any x_i. Locating a new service center in $[c_2, \infty)$ could change the nearest service center to x_2 or to x_3, but not to any $x_i < c_2$.

Thus, we would get a smaller value of the expression (2) by substituting $\min\{p_i \in S\}$ for c_*, so c_* cannot be a solution to the (k,m) WRMSDP. Similarly, if $c_* > \max\{p_i \in S\}$ then c_* cannot be a solution to the (k,m) WRMSDP. The assertion follows. □

Lemma 4.2 Suppose C is sorted in ascending order. Let $c_* \in [c_j, c_{j+1}]$, $p_i \notin [c_j, c_{j+1}]$. Then

$$\min\{l_1^1(p_i, y) \,|\, y \in C\} \; < \; l_1^1(p_i, c_*).$$

Proof: This follows from the observations that if $p_i < c_j \le c_* \le c_{j+1}$ then $|p_i - c_j| \le |p_i - c_*|$, and if $c_j \le c_* \le c_{j+1} < p_i$ then $|p_i - c_{j+1}| \le |p_i - c_*|$. □

See Figure 1 to help understand Lemma 4.2. Lemma 4.2 tells us that any positive contribution to the expression (3) for the reduction in the weighted total distance brought about by introducing the new service center c_* must be in terms with indices i such that $p_i \in [c_j, c_{j+1}]$ for some j, where $c_{-1} = -\infty$, $c_k = \infty$, and $c_* \in [c_j, c_{j+1}]$.

Lemma 4.3 For any nonnegative integer k, given an instance of the $(k,1)$ WRMSDP in R, there is a solution $c_* \in S$.

Proof: Assume the set C is sorted into ascending order. It follows from Lemma 4.2 that a solution c_* must satisfy $c_* \in [c_i, c_{i+1}]$ for some index i such that $S \cap [c_i, c_{i+1}] \neq \emptyset$. Therefore, we can take c_* to be a solution to the WRMSDP for the client set $S \cap [c_i, c_{i+1}]$. The assertion follows from Theorem 3.2. □

Theorem 4.4 Given an instance of the (k,m) WRMSDP in R, there is a solution set C_* such that $C_* \subset P$.

Proof: We argue by induction on the value of m. For $m = 1$, Lemma 4.3 yields the conclusion that we may take $C_* \subset P$.

Assume we may take $C_* \subset P$ for any instance of the (k,m) WRMSDP in R, for all integers $k > 0$, provided $m \leq u$ for some integer $u \geq 1$. Given an instance of the $(k,u+1)$ WRMSDP, by Lemma 4.3 there exists a solution set $C_* = \{c_{*,j}\}_{j=0}^{u}$ such that one of its members, say, $c_{*,0}$, is a member of P. Then, in order to have a min-sum of the weighted distances, we must have

$$C_* \setminus \{c_{*,0}\} \ = \ C_{v,l} \cup C_{v,r}$$

for some $v \in \{0,1,\ldots,u\}$, where

$$C_{v,l} = C_* \cap (-\infty, c_{*,0}), \ |C_{v,l}| = v, \ C_{v,r} = C_* \cap (c_{*,0}, \infty), \ |C_{v,r}| = u - v.$$

Then $C_{v,l}$ is a solution set for the instance of the (k,v) WRMSDP in R with client set $\{p_i \in P \mid p_i < c_{*,0}\}$ and existing service center set C, and $C_{v,r}$ is a solution set for the instance of the $(k,u-v)$ WRMSDP in R with client set $\{p_i \in P \mid p_i > c_{*,0}\}$ and existing service center set C. By the inductive hypothesis, we may assume $C_{v,l} \subset P$ and $C_{v,r} \subset P$. Therefore, $C_* \subset P$, as desired. $\square$

Notice in the above that we made no assumption about whether or not a member of C is a member of P.

Deriving analogs of Lemma 4.3 and Theorem 4.4 for higher dimensions must be done with care. For $d > 1$, easily constructed examples show that it need not be true that a new service center $c_* = (c_{*,1}, c_{*,2}, \ldots, c_{*,d})$ in a solution to the (k,m) WRMSDP may be taken from the members of P. Rather, what may be correctly derived is the much weaker conclusion that for each $j \in \{1,2,\ldots,d\}$, there exists $p_i \in S$ such that $c_{*,j} = p_{i,j}$.

In the following, the notation $C(n,m)$ represents the number of combinations of n objects taken m at a time. Recall

$$C(n,m) \ = \ \frac{n!}{m!(n-m)!}.$$

The running time of the algorithm in Theorem 4.5 may initially strike the reader as inefficient. It may be, however, that every subset T of m distinct members of P must be considered as a candidate for C_*, in which case the factor $C(n,m)$ may be necessary in analysis of an efficient solution for the (k,m) WRMSDP.

Theorem 4.5 Assume $0 \leq k$, $0 < m$, and $1 < k+m < n$. Then the (k,m) WRMSDP can be solved in R as follows.

- In $\Theta(nC(n,m))$ sequential time.
- In $\Theta(C(n,m))$ time using a CREW PRAM of n processors.

Proof: Consider the following algorithm.

(1) Sort each of S and C into ascending order. This step is performed in $\Theta(n\log n)$ sequential time or in $\Theta(\log n)$ time on the PRAM.

(2) By Theorem 4.4, there is a solution set $C_* \subset P$. Such a set C_* is computed as follows.

For each of the $C(n,m)$ subsets T of P such that $|T| = m$, do the following.

- Compute the ordered set $U = P \cup C \cup T$ by merge operations. This takes $\Theta(n)$ sequential time and can be performed in $O(\log n)$ time by the CREW PRAM, by using parallel binary searches for members of $C \cup T$ in P.
- For each $p_i \in T$, compute a nearest $q_i \in C \cup T$. This is done by performing two interval broadcasting operations in U, one so that each member of U learns its nearest preceding member of $C \cup T$ and the other so each member of U learns the nearest trailing member of $C \cup T$. Since interval broadcasts can be performed via parallel prefix operations [7], this step executes in $\Theta(n)$ sequential time or in $\Theta(\log n)$ time on the CREW PRAM.
- Compute

$$S_T = \sum_{i=0}^{n-1} w_i \min\{l_1(p_i, y) \,|\, y \in C \cup T\} = \sum_{i=0}^{n-1} w_i |p_i - q_i|.$$

This is done using a semigroup operation in $\Theta(n)$ sequential time or in $\Theta(\log n)$ time on the PRAM.

End For. Clearly, the sequential running time of this For loop is $\Theta(nC(n,m))$ and the time for the CREW PRAM is $\Theta(C(n,m))$.

(3) It follows from expression (2) that a solution C_* can be taken as a set T_* such that

$$S_{T_*} = \min\{S_T \,|\, T \subset P \text{ and } |T| = m\}.$$

This minimum can be computed by a semigroup operation in $\Theta(C(n,m))$ sequential time or in $\Theta(C(n,m)/n)$ time using the PRAM.

It follows that our solution is obtained in the running times asserted. $\square$

As noted above, a solution for the (k,m) WRMSDP in R^d can be obtained after solving in R for each component. However, applying our solution for R in each coordinate yields $C(n,m)$ candidates per coordinate, hence $[C(n,m)]^d$ points to consider as possible members of a solution set. This would yield $C([C(n,m)]^d,m)$ candidates for solution sets, each of which would require $\Theta(dm)$ time to evaluate in computing the minimum. Thus, the sequential running time for such an algorithm appears unimpressive, $\Theta(dmC([C(n,m)]^d,m))$.

5 Further remarks

We have given an algorithm for the Weighted Rectilinear Min-Sum Distance Problem in R^d. The algorithm is implemented on sequential computers and on a PRAM of n processors, where n is the number of client locations, and corrects the error of [10] in which it was wrongly assumed that the input is sorted in all dimensions. On both of these models of computation, the cost (product of running time and number of processors used) is $\Theta(dn\log n)$. Since the amount of input is $\Theta(dn)$, the algorithm is within a factor of $\log n$ of optimality.

We have generalized the WRMSDP and presented a solution for the (k,m) Weighted Rectilinear Min-Sum Distance Problem in R. The running times on both sequential computers and CREW PRAMs of n processors for our algorithm both include the large factor $C(n,m)$, corresponding in R to consideration of all subsets T of P such that $|T| = m$. It seems likely that in order to obtain a solution with asymptotically better running times, it would be necessary to show that the collection of subsets T of P that must be considered is much smaller, and that this collection can be determined efficiently.

Conflict of Interests

The author declares that there is no conflict of interests.

REFERENCES

[1] M. Chrobak, C. Kenyon, and N. Young, The reverse greedy algorithm for the metric *k*-median problem, Information Processing Letters 97 (2006), 68-72.

[2] R. Cole, Parallel merge sort, SIAM Journal on Computing 17 (4) (1988), 770-785.

[3] T. Drezner, Z. Drezner, and C.H. Scott, Location of a facility minimizing nuisance to or from a planar network, Computers & Operations Research 36 (2009), 135-148.

[4] R.Z. Farahani, Z. Drezner, and N. Asgari, Single facility location and relocation problem with time dependent weights and discrete planning horizon, Annals of Operations Research 167 (2009), 353-368.

[5] S.L. Hakimi, Optimum locations of switching centers and the absolute centers and medians of a graph, Operations Research 12 (1964), 450-469.

[6] S.L. Hakimi, Optimum distribution of switching centers in a communication network and some related graph-theoretic problems Operations Research 13 (1965), 462-475.

[7] R. Miller and L. Boxer, Algorithms Sequential & Parallel: A Unified Approach, 3rd ed., Cengage Learning, Boston, 2013.

[8] A.P. Punnen and Y.P. Aneja, On *k*-sum optimization, Operations Research Letters 18 (1996), 233-236.

[9] A. Tamir, Sorting weighted distances with applications to objective function evaluations in single facility location problems, Operations Research Letters 32 (2004), 249-257.

[10] D. Zhu and X. Wang, An optimal algorithm for the weighted median problem, Journal of Computers 9 (2) (2014), 257-265.