Operations Research Letters 10 (1991) 395–402
North-Holland

October 1991

# Improved complexity bounds for location problems on the real line

Refael Hassin *, Arie Tamir

*Department of Statistics, Raymond and Beverly Sackler, Faculty of Exact Sciences, Tel Aviv University, Ramat-Aviv, Tel Aviv 69978, Israel*

Received May 1990
Revised August 1990

In this note we apply recent results in dynamic programming to improve the complexity bounds of several median and coverage location models on the real line.

facility location, center problems, median problems, computational complexity

## 0. Introduction

We consider a general facility location model on the real line, where we have to select $p$ sites for facilities to serve $n$ demand points. The model extends and unifies some of the classical problems in location theory, e.g., the $p$-median problem, the $p$-coverage problem, and the simple uncapacitated plant location problem. We apply recent results in dynamic programming [6,8,15] to improve the complexity bounds of the general model from $O(pn^2)$ to $O(n^2)$. For the linear cases – the $p$-median and the simple plant location models – we obtain $O(pn)$ and $O(n)$ bounds, respectively, by implementing the elegant computational geometry approach of [14]. Finally, for the stepwise case – the $p$-coverage problem – we present a parametric approach which yields a further improvement under additional assumptions.

## 1. The general model

Let $N = \{v_1, \ldots, v_n\}$ be a set of $n$ points on the real line, identified as the set of demand points. In a location model the objective is to select a subset $S \subseteq N$ of points that will serve all the demand points in $N$. The points in $S$ are identified as the supply points or service centers. We assume that all potential centers provide identical services and they are uncapacitated. The utility of a demand point $v_i$ depends on its distance to the nearest center. Specifically, for $i = 1, \ldots, n$, we let $f_i(d(v_i, S))$ be the real monotone nondecreasing (disutility) function of $v_i$, where

$$d(v_i, S) = \text{Min}\{\, |v - v_i| : v \in S \}.$$

For $S = \{v\}$ we denote $d(v_i, S)$ by $d(v_i, v)$. We assume that $f_i(0) = 0$, $i = 1, \ldots, n$. For $i = 1, \ldots, n$, let $C_i$ be the fixed setup cost of establishing a center at point $v_i$. We also assume that there is an upper bound, $p$, on the total number of centers that can be selected in $N$. We are concerned with the following location model,

$$\begin{aligned} \underset{S}{\text{Minimize}} \quad & \sum_{i \in S} C_i + \sum_{i=1}^{n} f_i(d(v_i, S)) \\ \text{subject to} \quad & S \subseteq N, \\ & |S| \leq p. \end{aligned} \tag{1.1}$$

The above formulation unifies several location models in the literature. For example, when $C_i = 0$, and $f_i$ is linear for each $i = 1, \ldots, n$, (1.1) reduces to the well known $p$-median model on the line. When $f_i$, $i = 1, \ldots, n$, is a step function

$$f_i(x) = \begin{cases} 0 & \text{if } x \le r_i, \\ b_i & \text{otherwise,} \end{cases} \tag{1.2}$$

we obtain the $p$-coverage location problem of [2,3,7,11].

If the constraint $|S| \le p$ is redundant, e.g. $p \ge n$, (1.1) coincides with the simple uncapacitated plant location problem [9] on the line.

To solve (1.1) we suggest the two function recursive approach. Suppose that $v_1 < v_2 < \cdots < v_n$. Let $N^j = \{v_j, \ldots, v_n\}$, $j = 1, \ldots, n$, and consider the following set of subproblems. Define, for $j = 1, \ldots, n$, and $q \le p$,

$$F^q(j) = \operatorname*{Min}_{S \subseteq N^j} \left\{ \sum_{i \in S} C_i + \sum_{i=j}^{n} f_i(d(v_i, S)) : |S| \le q \right\},$$

$$G^q(j) = \operatorname*{Min}_{S \subseteq N^j} \left\{ \sum_{i \in S} C_i + \sum_{i=j}^{n} f_i(d(v_i, S)) : |S| \le q,\ v_j \in S \right\}.$$

$F^q(j)$ is the optimal objective value of the subproblem reduced to $N^j$, where $q$ is the upper bound, and $G^q(j)$ is the respective value for the same subproblem, provided that a center must be set at $v_j$.

With the above definition we obtain the two recursions,

$$G^q(j) = C_j + \operatorname*{Min}_{j < k \le n+1} \left\{ \sum_{t=j}^{k-1} f_t\big(d(v_t, v_j)\big) + F^{q-1}(k) \right\}, \quad q \ge 2,\ j = 1, \ldots, n, \tag{1.3a}$$

$$F^q(j) = \operatorname*{Min}_{j \le k \le n} \left\{ \sum_{t=j}^{k} f_t\big(d(v_t, v_k)\big) + G^q(k) \right\}, \quad q \ge 1,\ j = 1, \ldots, n. \tag{1.3b}$$

The boundary conditions are

$$G^1(j) = C_j + \sum_{t=j}^{n} f_t\big(d(v_t, v_j)\big), \quad j = 1, \ldots, n,\ F^q(n+1) = 0,\ q \ge 1. \tag{1.4}$$

The solution value for (1.1) is then given by $F^p(1)$. Assuming that it takes constant time to evaluate $f_t(x)$ for any $t = 1, \ldots, n$, and argument $x$, $F^p(1)$ can be computed in $O(pn^2)$ time, using (1.3) and (1.4). If the constraint $|S| \le p$ in (1.1) is known to be redundant, we obtain the recursive equations

$$G(j) = C_j + \operatorname*{Min}_{j < k \le n+1} \left\{ \sum_{t=j}^{k-1} f_t\big(d(v_t, v_j)\big) + F(k) \right\}, \quad j = 1, \ldots, n-1, \tag{1.5a}$$

$$F(j) = \operatorname*{Min}_{j \le k \le n} \left\{ \sum_{t=j}^{k} f_t\big(d(v_t, v_k)\big) + G(k) \right\}, \quad j = 1, \ldots, n, \tag{1.5b}$$

$$G(n) = C_n, \qquad F(n+1) = 0, \tag{1.5c}$$

for the respective subproblems. Therefore, $F(1)$, the optimal value for this case, can be computed in $O(n^2)$ time. We will refer to the case when the constraint $|S| \le p$ is redundant as the relaxed model.

We will first show how to implement the recent results in [1,6,8,15] to improve upon the above complexity bounds. We will then focus on the cases where the disutilities are linear or step functions, and obtain further simplifications.

To show the applicability of the algorithms in [1,6,8,15] to our model we need the following definition and lemma.

For $1 \leq j \leq k \leq n$, define

$$w(j, k) = \sum_{t=j}^{k} f_t(d(v_t, v_j)),$$

$$\bar{w}(j, k) = \sum_{t=j}^{k} f_t(d(v_t, v_k)).$$

**Lemma 1.** *Let $j$, $k$, $l$ and $m$ be four indices satisfying $1 \leq j \leq k \leq l \leq m \leq n$. Then,*

$$w(j, m) - w(j, l) \geq w(k, m) - w(k, l), \tag{1.6}$$

$$\bar{w}(j, m) - \bar{w}(j, l) \geq \bar{w}(k, m) - \bar{w}(k, l). \tag{1.7}$$

**Proof.**

$$w(j, m) - w(j, l) = \sum_{t=l+1}^{m} f_t(d(v_t, v_j)) \geq \sum_{t=l+1}^{m} f_t(d(v_t, v_k)) = w(k, m) - w(k, l)$$

where the inequality follows from the monotonicity of the functions $f_t$, $t = 1, \ldots, n$. Similarly,

$$\bar{w}(j, m) - \bar{w}(k, m) = \sum_{t=j}^{k-1} f_t(d(v_t, v_m)) \geq \sum_{t=j}^{k-1} f_t(d(v_t, v_l)) = \bar{w}(j, l) - \bar{w}(k, l).$$

The inequalities (1.6) and (1.7) are recognized as the quadrangle or concavity property in [1,6,8,15]. They can also be viewed as the supermodularity property when $w$ and $\bar{w}$ are regarded as functions defined on the collection of intervals on the real line.

With the above definition we rewrite (1.3) as

$$G^q(j) = C_j + \operatorname*{Min}_{j<k \leq n+1} \{w(j, k-1) + F^{q-1}(k)\},$$

$$F^q(j) = \operatorname*{Min}_{j \leq k \leq n} \{\bar{w}(j, k) + G^q(k)\}.$$

Using the above concavity property and assuming that $w(j, k)$ and $\bar{w}(j, k)$ can be computed in constant time for a fixed pair of indices $j < k$, we can use the algorithms in [6,8,15] to solve (1.3). In particular, the algorithms in [6,8,15] will yield $G^q(j)$ and $F^q(j)$ for all $q \leq p$ and $j = 1, \ldots, n$, in $O(pn)$ total effort. Similarly, $G(j)$ and $F(j)$ in (1.5) for all $j = 1, \ldots, n$, can be computed in a total of $O(n)$ time.

It is certainly obvious that it takes $O(n^2)$ time to compute $w(j, k)$ and $\bar{w}(j, k)$ for all indices $1 \leq j \leq k \leq n$. Therefore, the solution value, $F^p(1)$, for (1.1) can be obtained in $O(pn + n^2) = O(n^2)$ time.

Next, we focus on some special cases of the functions $f_t$, $t = 1, \ldots, n$, and obtain further improvements. These improvements will follow from subquadratic preprocessing that will enable an efficient computation of $w(j, k)$ and $\bar{w}(j, k)$ for a fixed pair $j < k$.

## 2. The linear case

Suppose that for each $t = 1, \ldots, n$, $f_t(x) = a_t x$ for some nonnegative $a_t$. Then for $1 \leq j < k \leq n$ we have

$$w(j, k) = \sum_{t=j}^{k} a_t(v_t - v_j) = \sum_{t=1}^{k} a_t v_t - \sum_{t=1}^{j-1} a_t v_t - \left(\sum_{t=j}^{k} a_t\right) v_j,$$

$$\bar{w}(j, k) = \sum_{t=j}^{k} a_t(v_k - v_t) = -\sum_{t=1}^{k} a_t v_t + \sum_{t=1}^{j-1} a_t v_t + \left(\sum_{t=j}^{k} a_t\right) v_k.$$

Define, for $j = 1, \ldots, n$,

$$A(j) = \sum_{t=1}^{j} a_t, \qquad \mathrm{AV}(j) = \sum_{t=1}^{j} a_t v_t.$$

Now,

$$w(j, k) = \mathrm{AV}(k) - \mathrm{AV}(j-1) - (A(k) - A(j-1)) v_j, \tag{2.1a}$$

$$\bar{w}(j, k) = -\mathrm{AV}(k) + \mathrm{AV}(j-1) + (A(k) - A(j-1)) v_k. \tag{2.1b}$$

It follows from (2.1) that after some preprocessing that consumes $O(n)$ time, $w(j, k)$ and $\bar{w}(j, k)$ can be obtained in constant time for a fixed pair $j \le k$. Therefore (1.3) can be solved in $O(pn)$ time for the linear case. When the constraint $|S| \le p$ is redundant the bound reduces to $O(n)$.

Simpler algorithms with the same bounds can be derived by applying the simple and elegant ideas in [14]. To facilitate the discussion consider for example, the equations in (1.5). Using (2.1) we rewrite (1.5) as

$$G(j) = C_j + \min_{j<k\le n+1} \{ -\mathrm{AV}(j-1) + A(j-1) v_j - A(k-1) v_j + \mathrm{AV}(k-1) + F(k) \},$$

$$F(j) = \min_{j\le k\le n} \{ \mathrm{AV}(j-1) - A(j-1) v_k + A(k) v_k - \mathrm{AV}(k) + G(k) \}.$$

Define, for $j = 1, \ldots, n$,

$$C_j' = C_j - \mathrm{AV}(j-1) + A(j-1) v_j,$$

$$C_j'' = \mathrm{AV}(j-1),$$

$$F'(j) = F(j) + \mathrm{AV}(j-1),$$

$$G'(j) = G(j) - \mathrm{AV}(j) + A(j) v_j.$$

Then,

$$G(j) = C_j' + \min_{j<k\le n+1} \{ -A(k-1) v_j + F'(k) \}, \tag{2.2a}$$

$$F(j) = C_j'' + \min_{j\le k\le n} \{ -A(j-1) v_k + G'(k) \}. \tag{2.2b}$$

To solve the two recursive equation system (2.2) we modify and implement the idea used in [14] to solve a single recursive equation system. For the sake of completeness we briefly review that idea. Consider the equation defining $G(j)$. For each $k > j$ consider the point in the plane with coordinates $(A(k-1), F'(k))$. A minimizer, $k = k(j)$, in the equation defining $G(j)$ corresponds to an extreme point in the convex hull of the planar set of points $(A(k-1), F'(k))$, $j < k \le n+1$. Thus, it is sufficient to maintain only these extreme points. The minimizer amongst those points is determined by the coefficient $-v_j$. This coefficient is monotone in $j$ and therefore we can assume that the sequence of minimizers is monotone, i.e., $k(j) \le k(j+1)$. A similar observation holds for the second equation defining $F(j)$, where we look at the set of points in the plane, $(v_k, G'(k))$, $j \le k \le n$. These monotonicity properties of the minimizers enable us to use standard techniques from computational geometry [4,13] to efficiently construct and maintain the two convex hulls as $j$ varies from $n$ to 1.

All the values $G(j)$ and $F(j)$, $j = 1, \ldots, n$, are generated in this process in linear time. The linkage between the two convex hulls is governed by the two equations of (2.2). Specifically, we start with $G(n) = F(n) = C_n$. Then, recursively for $j = n-1, \ldots, 1$ we first compute $G(j)$ and then evaluate $F(j)$.

## 3. The stepwise case

Consider the case where each $t = 1, \ldots, n$ is associated with the two positive numbers $r_t$ and $b_t$ and

$$f_t(x) = \begin{cases} 0 & \text{if } x \le r_t, \\ b_t & \text{otherwise.} \end{cases}$$

This model is the $p$-coverage problem discussed in [2,3,7,11]. As an instance of (1.1) it is solvable in $O(pn + n^2) = O(n^2)$ time. (Note that the bound for the relaxed model is also $O(n^2)$.) We will show a different algorithm for this model which is based on a parametric approach. This algorithm will solve the relaxed model in $O(n \log n)$ time and the general case in $O(n^2 \log^2 n)$ time.

Before we present the parametric approach we treat two special subcases by the general method discussed above.

**Subcase I.** $b_t = \infty$ for $t = 1,\dots, n$. It is easy to verify that $w(j, k)$ and $\bar{w}(j, k)$ are now given by

$$w(j, k) = \begin{cases} 0 & \text{if } |v_j - v_t| \le r_t \text{ for } j \le t \le k, \\ \infty & \text{otherwise,} \end{cases}$$

$$\bar{w}(j, k) = \begin{cases} 0 & \text{if } |v_t - v_k| \le r_t \text{ for } j \le t \le k, \\ \infty & \text{otherwise.} \end{cases}$$

For $j = 1,\dots, n$, let $k(j)$ ($\bar{k}(j)$) be the largest index $k \ge j$ such that $w(j, k) = 0$ ($\bar{w}(j, k) = 0$). The sequences $\{k(j)\}$ and $\{\bar{k}(j)\}$ are both monotone nondecreasing in $j$. Therefore, both sequences can be generated in $O(n)$ time.

With the above notation we obtain for $j \le k$

$$w(j, k) = \begin{cases} 0 & \text{if } k \le k(j), \\ \infty & \text{otherwise,} \end{cases}$$

$$\bar{w}(j, k) = \begin{cases} 0 & \text{if } k \le \bar{k}(j), \\ \infty & \text{otherwise.} \end{cases}$$

For a given pair $(j, k)$, $j < k$, both $w(j, k)$ and $\bar{w}(j, k)$ are computable in constant time. Therefore, for this subcase (1.3) and (1.5) are solved in $O(pn)$ and $O(n)$ times, respectively.

**Subcase II.** $r_t = r$ for $t = 1,\dots, n$. For each $j$, $j = 1,\dots, n$, define $l(j)$ ($\bar{l}(j)$) to be the largest (smallest) index $l$ ($\bar{l}$) such that $|v_l - v_j| \le r$ ($|v_{\bar{l}} - v_j| \le r$). The sequences $\{l(j)\}$ and $\{\bar{l}(j)\}$ are both monotone and therefore can be computed in $O(n)$ time. Defining for $t = 1,\dots, n$, $B_t = \sum_{i=1}^{t} b_i$, we obtain for $j \le k$,

$$w(j, k) = \begin{cases} 0 & \text{if } k \le l(j), \\ B_k - B_{l(j)} & \text{if } k > l(j), \end{cases}$$

$$\bar{w}(j, k) = \begin{cases} 0 & \text{if } j \ge \bar{l}(k), \\ B_{\bar{l}(k)-1} - B_{j-1} & \text{if } j < \bar{l}(k). \end{cases}$$

The total effort to compute $B_t$, $t = 1,\dots, n$, is $O(n)$. Since $w(j, k)$ and $\bar{w}(j, k)$ can be computed in constant time for a fixed pair $(j, k)$ we conclude that (1.3) and (1.5) can be solved in $O(pn)$ and $O(n)$ times, respectively.

## 4. A parametric approach for the stepwise case

We have noted above that this model can be solved in $O(pn + n^2) = O(n^2)$ time using the general method. As explained above, the bottleneck term, $n^2$, accounts only for the preprocessing phase when we compute $w(j, k)$ and $\bar{w}(j, k)$ for all indices $1 \le j < k \le n$. Because of that the $O(n^2)$ bound applies also to the relaxed case where the constraint $|S| \le p$ in (1.1) is omitted. We will now present a different solution procedure for the $p$-coverage model based on a parametric approach, which solves the problem in $O(n^2 \log^2 n)$ time and the relaxed case in $O(n \log n)$ time. Let $A = (a_{ij})$ be an $n \times n$ 0–1 matrix where

$$a_{ij} = \begin{cases} 1 & \text{if } |v_i - v_j| \leqslant r_i, \\ 0 & \text{otherwise.} \end{cases}$$

The $p$-coverage problem is

$$\text{Min} \quad \sum_{j=1}^{n} C_j x_j + \sum_{i=1}^{n} b_i z_i \tag{4.1}$$

$$\text{s.t.} \quad (Ax)_i + z_i \ge 1, \quad i=1,\dots,n,$$

$$\sum_{j=1}^{n} x_j \le p,$$

$$x_j \in \{0,1\}, \quad j=1,\dots,n,$$

$$z_i \in \{0,1\}, \quad i=1,\dots,n.$$

The matrix $A$ has the row consecutive 1's property. Therefore, the binary constraints on the variables can be relaxed and replaced by a nonnegative requirement without affecting the optimal solution [12]. Taking the Lagrangian with respect to the constraint $\sum_{j=1}^{n} x_j \le p$, (4.1) is equivalent to

$$\underset{\lambda \le 0}{\text{Maximize}} \{\lambda p + g(\lambda)\} \tag{4.2}$$

where

$$g(\lambda) = \text{Min} \sum_{j=1}^{n} (C_j - \lambda) x_j + \sum_{i=1}^{n} b_i z_i$$

$$\text{s.t.} \quad (Ax)_i + z_i \ge 1, \quad i=1,\dots,n,$$

$$x,\ z \ge 0.$$

Note that $g(\lambda)$ is a concave, monotone nonincreasing and piecewise linear function of $\lambda$.

For each $\lambda$, for which $g(\lambda)$ is finite, $g(\lambda)$ is attained at some 0–1 vector $x$. Therefore, the slopes of $g(\lambda)$ constitute a decreasing sequence of nonpositive integers, where the smallest (finite) slope is bounded below by $-n$. In particular, $g(\lambda)$ has at most $n$ breakpoints. The maximum value at (4.2) is attained at a breakpoint, say $\lambda^*$, satisfying

$$-g'(\lambda^* -) < p \le -g'(\lambda^* +)$$

where $g'(\lambda^* -)$ and $g'(\lambda^* +)$ are, respectively, the left and right derivatives of $g(\lambda)$ at $\lambda = \lambda^*$. Let $T(n)$ be the computational effort required to compute $g(\lambda)$ for a given value of $\lambda$. (Note that $g(0)$ is the solution value to the relaxed model). Then using the parametric algorithm in [10], $\lambda^*$ can be obtained in $O((T(n))^2)$ time. We will show that $T(n) = O(n \log n)$. That would imply an $O(n^2 \log^2 n)$ algorithm for the $p$-coverage problem.

We consider the following coverage model

$$\text{Min} \quad \sum_{j=1}^{n} \bar{C}_j x_j + \sum_{i=1}^{n} b_i z_i \tag{4.3}$$

$$\text{s.t.} \quad (Ax)_i + z_i \ge 1, \quad i=1,\dots,n,$$

$$x \ge 0,\ z \ge 0$$

where $\bar{C}_j$, $b_j \ge 0$, $j=1,\dots,n$. It should be noted that the matrix A, which has the row consecutive 1's property, is stored as follows. For each $i$, there exist integers $n_i \le m_i$ such that

$$q_{ij} = \begin{cases} 1 & \text{if } n_i \le j \le m_i, \\ 0 & \text{otherwise.} \end{cases}$$

Thus, for each row index $i$, $i=1,\dots,n$, we store its pair of indices $n_i$ and $m_i$ only. We first permute (in $O(n)$ time) the rows of $A$ so that the permuted matrix, say $\bar{A}$, satisfies the following rule:

For every pair of rows $i$ and $k$ in $\bar{A}$,

$$i > k \quad \text{implies} \quad m_i \ge m_k.$$

It is easy to verify that $\bar{A}$ does not have $\left[\begin{smallmatrix}1&1\\1&0\end{smallmatrix}\right]$ as a $2\times 2$ submatrix. A matrix with this property is called a *greedy matrix in standard form* [9]. The dual of (4.3) is

$$\begin{aligned} \text{Max} \quad & \sum_{i=1}^{n} y_i \\ \text{s.t.} \quad & (\bar{A}'y)_j \le \bar{C}_j, \quad j=1,\dots,n, \\ & y_i \le b_i, \quad i=1,\dots,n, \\ & y \ge 0. \end{aligned} \tag{4.4}$$

The matrix $\bar{A}'$ possesses two essential properties:

(1) It is in greedy standard form.

(2) It has the column consecutive 1's property.

Therefore, from [9], (4.4) can be solved by the following greedy algorithm.

**Greedy Algorithm.** Consider $y_i$, $i=1,\dots,n$, in (4.4), by increasing index of $i$, and set it equal to the largest possible value with respect to the constraints. Thus,

$$y_1 = \text{Min}\left\{b_1, \text{Min}\left\{\bar{C}_j : n_1 \le j \le m_1\right\}\right\},$$

and for $1 < i \le n$,

$$y_i = \text{Min}\left\{b_i, \text{Min}\left\{\bar{C}_j - \sum_{k=1}^{i-1} y_k a_{kj} : n_i \le j \le m_i\right\}\right\}.$$

We claim that the above greedy procedure can be implemented in $O(n \log n)$ time. Initially, we are given a set of $n$ numbers $\bar{C}_1, \bar{C}_2, \dots, \bar{C}_n$ which are updated at each one of the $n$ steps. At step $i$ of the algorithm we are given an interval of indices, $\{j : n_i \le j \le m_i\}$, and we find the minimum of the current values of these numbers over this interval. Let $\alpha_i$ denote this minimum. We then subtract $\delta_i = \text{Min}\{b_i, \alpha_i\}$ from each number in this interval and proceed to the next step.

There are simple data structures that require $O(n)$ preprocessing time and then perform each one of the two typical operations ('minimum over an interval' and 'subtract over an interval') in $O(\log n)$ time. The reader is referred to [5] for more details concerning such data structures.

To conclude we have shown that the parametric approach can be used to solve the general $p$-coverage model in $O(n^2 \log^2 n)$ time. Also, if the constraint $e'x \le p$ in (4.1) is redundant the effort is only $O(n \log n)$, since only $g(0)$ has to be computed.

## References

[1] A. Aggarwal and J.K. Park, "Applications of array searching to economic lot-size problems," Technical Report, IBM Research Division, T.J. Watson Research Center, Yorktown Heights, NY, 1989.

[2] M.W. Broin and T.J. Lowe, "A dynamic programming algorithm for covering problems with (greedy) totally balanced constraint matrices", *SIAM J. Algebraic Discrete Methods* 7 (1986) 348–357.

[3] M. Daskin, P.C. Jones and T.J. Lowe, "Rationalizing tool selection in flexible manufacturing system for sheet-metal products", to appear in *Oper. Res.*

[4] H. Edelsbrunner, *Algorithms in Combinatorial Geometry*, Springer-Verlag, Heidelberg, Germany, 1987.

[5] Z. Galil, S. Micali and H. Gabow, "Priority queues with variable priority and an $O(EV \log V)$ algorithm for finding a maximal weighted matching in general graphs", *SIAM J. Comput.* **15** (1986) 120–130.

[6] Z. Galil and K. Park, "A linear time algorithm for concave one-dimensional dynamic programming", Unpublished manuscript, Department of Computer Science, Columbia University, 1989.

[7] W. Hsu, "The distance-domination numbers of trees", *Oper. Res. Lett.* **1** (1982) 96–100.

[8] M.M. Klawe, "A simple linear time algorithm for concave one-dimensional dynamic programming", Technical Report 89-16, University of British Columbia, Vancouver, 1989.

[9] A. Kolen and A. Tamir, "Covering problems", in: P. Mirchandani and R.L. Francis (eds.) *Discrete Location Theory*, Wiley, New York, 1990.
[10] N. Megiddo, "Combinatorial optimization with rational objective function", *Math. Oper. Res.* **4** (1979) 414–424.
[11] N. Megiddo, E. Zemel and S.L. Hakimi, "The maximum coverage location problem", *SIAM J. Algebraic Discrete Methods* **4** (1983) 253–261.
[12] G.L. Nemhauser and L.A. Wolsey, *Integer and Combinatorial Optimization*, Wiley, New York, 1988.
[13] F.P. Preparata and M.I. Shamos, *Computational Geometry*, Springer-Verlag, New York, 1985.
[14] A.P.M. Wagelmans, C.P.M. Van Hoesel and A.W.J. Kolen, "Economic lot-sizing: An O($n$ log $n$)-algorithm that runs in linear time in the Wagner–Whitin case", Report 8952/A, Erasmus University, Rotterdam, 1989.
[15] R. Wilber, "The concave least-weight subsequence problem revisited", *J. Algorithms* **9** (1988) 418–425.