RAIRO. RECHERCHE OPÉRATIONNELLE

B. PELEGRÍN
L. CÁNOVAS

New heuristic algorithms for the rectangular p-cover problem

RAIRO. Recherche opérationnelle, tome 29, nº 1 (1995), p. 73-91
<http://www.numdam.org/item?id=RO_1995__29_1_73_0>

Recherche opérationnelle/Operations Research
(vol. 29, n° 1, 1995, p. 73 à 91)

NEW HEURISTIC ALGORITHMS FOR THE RECTANGULAR p-COVER PROBLEM (*)

by B. PELEGRÍN [1] and L. CÁNOVAS [1]

Communicated by Brian BOFFEY

Abstract. – *Many heuristic algorithms have been proposed in the literature for the solution of p-center problems, most of which can be used in any metric space. However, little computational experience has been reported with these heuristics, most times for problems in $\mathbb{R}^2$ with the Euclidean distance. In this paper, we consider the unweighted p-center problem in $\mathbb{R}^m$ when distance is measured by the Tchebycheff norm, which we name the Rectangular p-Cover Problem. We propose new heuristic algorithms for this problem, and present computational results. Firstly, a new class of heuristics based on the generation of seed points is given, which is obtained using a new assignment rule. Secondly, two new algorithms based on partitions are given, which can be seen as heuristics of improvement type. Finally, it is shown by computational experiments that the new algorithms improve some other related heuristics considered in this paper.*

Keywords: Clustering, Tchebycheff norm, unweighted p-center.

Résumé. – *Beaucoup d'algorithmes heuristiques ont été proposés en littérature pour la solution du problème p-centre, la plupart d'eux peut être utilisée dans n'importe quel espace métrique. Du fait, peu d'expériences de computation ont été élaborées avec ces heuristiques, la plupart du temps en $\mathbb{R}^2$ avec la distance Euclidienne. Dans cet article, on considère les problèmes dans $\mathbb{R}^m$ du p-centre sans poids quand la distance mesurée par la norme de Tchebycheff, qui s'appelle le Problème dep-Couvrement Rectangulaire. On suppose de nouveaux algorithmes heuristiques pour ce problème, et on présente des résultats de computation. Premièrement, une nouvelle classe d'heuristiques basées sur l'engendrement des points de semence donné, qui est obtenue en utilisant une nouvelle règle d'allocation. Deuxièmement, deux nouveaux algorithmes basés dans la répartition sont donnés, lesquels peuvent être vus comme une amélioration des heuristiques. Finalement, on montre par expérience de computation que les nouveaux algorithmes améliorent certaines heuristiques considérées dans cet article.*

Mots clés : Amas, norme de Tchebycheff, p-centre sans poids.

1. INTRODUCTION

Let $M = \{P_1, ..., P_n\}$ be a finite set of points in the Euclidean space $\mathbb{R}^m$. It is often required in Location and Cluster Analysis to partition the set M into p subsets $M_1, ..., M_p$, assuming $M_j \neq \emptyset$, $M_j \cap M_t = \emptyset$, $\cup M_j = M$,

(*) Received March 1994.

[1] Dpto. Matematica Aplicada y Estadistica, Universidad de Murcia, Spain.

so that a given function $g(M_1, \ldots, M_p)$ will be minimized. This problem is known as the *unweighted p-center problem* when the function g is the maximum among the radii of the subsets. The radius of a subset M_j is defined as the optimal value $F(M_j)$ of the following 1-center problem:

$$\underset{x \in \mathbb{R}^m}{\text{Minimize}} \text{ Maximum } \{d(P_i, X) : P_i \in M_j\} \qquad \mathbf{(1CM_j)}$$

where $d(.,.)$ is a given metric in $\mathbb{R}^m$. An optimal solution C_j of $(\mathbf{1CM}_j)$ is called a center of M_j. The unweighted p-center problem is then formulated as:

$$\underset{\alpha \in P(M, p)}{\text{Minimize}} \; F_\alpha = \text{Maximum } \{F(M_1), \ldots, F(M_p)\} \qquad \mathbf{(pC)}$$

where $P(M, p)$ denotes the set of all partitions of M into p disjoint subsets.

In Location, to mention a few examples, one may refer to the location of fire stations, ambulance bases, police stations and messenger delivery services. Usually, the points P_i represent incidents, destinations, users, clients etc.; the centers C_j represent facilities which satisfy demand of points in M_j; and $d(P_i, C_j)$ represent distance or travel time between P_i and C_j. In Cluster Analysis, this type of problem may occur in almost all empirically based disciplines such as economics, engineering, biology, archaeology, social sciences, etc. Then, the points P_i represent data, objects or entities, which are characterized by the values of m variables; the subsets M_j represent groups, each of them is as homogeneous as possible; each center C_j is an ideal point which represents the points in group M_j; and $d(P_i, C_j)$ is a measure of dissimilarity between points P_i and C_j.

We define the *rectangular p-cover problem* as to find p hyperrectangles, with the sides parallel to the axes, containing the points in M so that to minimize the maximum length of their sides. As an illustration in the plane, consider the following regional planning problem: given a set of undivisible areas (represented by points in $\mathbb{R}^2$) constituting a geographical or urban region, cluster these areas into p rectangles with minimum length of their sides, in such a way that a source of a certain social service (as fire stations, ambulance bases, schools, police stations, etc.) will be located at the center of each rectangle to service the areas in it contained. To formulate this problem we will use the following notation:

$$R = \{x = (x_1, \ldots, x_m) \in \mathbb{R}^m : a_k \leq x_k \leq b_k, \; k = 1, \ldots, m\}$$

an hyperrectangle,

$$L = \operatorname{Max}\{b_k - a_k : k = 1, \ldots, m\}$$

the maximum length of a side of R,

$$x_k(P)$$

the k-th coordinate of the point P.

Given a subset M_j of M the smallest hyperrectangle containing M_j is given by

$$R(M_j) = \{x = (x_1, \ldots, x_m) \in \mathbb{R}^m : a_k^j \leq x_k \leq b_k^j,\ k = 1, \ldots, m\}$$

where $a_k^j = \text{Minimum}\{x_k(P) : P \in M_j\}$ and $b_k^j = \text{Maximum}\{x_k(P) : P \in M_j\}$. Let $L(M_j)$ be the maximum lengtht of a side of $R\ (M_j)$, *i.e.*, $L(M_j) = \text{Maximum}\{b_k^j - a_k^j : k = 1, \ldots, m\}$. Then the rectangular p-cover problem can be formulated as:

$$\underset{\alpha \in P(M,p)}{\text{Minimize}}\, L_\alpha = \text{Maximum}\{L(M_1), \ldots, L(M_p)\} \qquad \textbf{(RpC)}$$

The problem **(RpC)** can be seen as the problem **(pC)** when the metric is given by the Tchebycheff norm, *i.e.*,

$$d(P_i, C_j) = \operatorname{Max}\{|x_k(P_i) - x_k(C_j)| : k = 1, \ldots, m\}$$

since then it is verified that

$$F(M_j) = 1/2 \max\{d(P_i, P_h) : P_i, P_h \in M_j\} = 1/2\, L(M_j)$$

[13], and therefore $F_\alpha = 1/2\, L_\alpha$. For $m = 2$, **(RpC)** can also be seen as **(pC)** when the metric is given by the Rectangular norm, *i.e.*,

$$d(P_i, C_j) = |x_1(P_i) - x_1(C_j)| + |x_2(P_i) - x_2(C_j)|$$

since in $\mathbb{R}^2$ the Tchebycheff norm is obtained from the Rectangular norm taking the transformation $y_1 = x_1 + x_2, y_2 = x_1 - x_2$ (then $|x_1| + |x_2| = \max\{|y_1|, |y_2|\}$). This means that rectangles whose sides determine angles of 45° with the axes (x-coordinates) become rectangles with sides parallel to the axes (y-coordinates) [10].

The problem **(pC)** has been proved to be NP-Hard, even for $m = 2$ and the Rectangular norm [11, 12], and so only heuristic algorithms can be used to obtain good solutions of **(RpC)** for large problems [2, 6, 7, 14, 16]. Most of the heuristic algorithms proposed for **(pC)** can be used for any metric [14], and consequently for the problem **(RpC)**. Many of them require the corresponding 1-center problem to be solved a lot of times, which is generally time consuming for $m > 2$. Some algorithms to evaluate $F(M_j)$ for different metrics can be found in [4, 8, 10] for $m = 2$, and in [3, 5, 9, 13] for any m. Computational experience has also been reported for **(pC)**, but it is limited to points in $\mathbb{R}^2$ and not very large values of n and p, for instance $n \in [20, 150]$, $p \in [2, n-1]$ in [7], and $n \leq 2\,000$, $p \leq 10$ in [2].

In this paper, we propose new heuristic algorithms for the solution of **(RpC)** which are related to some heuristics for **(pC)**. In section 2, we consider a class of heuristics for the solution of **(RpC)**, for which a new assignment rule is given. With the new rule each algorithm is this class becomes a new algorithm, creating a new class of algorithms. In particular, modifications of two algorithms given in [6, 16] are considered. In section 3, we give two new algorithms based on partitions, the first is a modification of the heuristic given in [13], and the second is a modification of the well known Location-Allocation algorithm, which requires the evaluation of $F(M_j)$ many times, but this is not time consuming for $F(M_j) = L(M_j)$, *i.e.*, for **(RpC)**. In section 4, we show by computational experiments that the new heuristics improve the mentioned existing heuristics when they are used for the problem **(RpC)**.

2. ALGORITHMS BASED ON SEED POINTS

Some of the existing heuristic algorithms for **(pC)** are based on the generation of a set of p seed points. A class of these algorithms, when they are applied to obtain a solution of **(RpC)**, can be described as follows:

Class of seed point algorithms for (RpC)

Step 1: Generate a set of p seed points $C_1, \ldots, C_p$ in $\mathbb{R}^m$.

Step 2: Obtain a partition $\alpha = \{M_1, \ldots, M_p\}$ assigning each point P_i to its closest seed point, *i.e.*,

$$M_j = \{P_i \in M : d(P_i, C_j) \leq d(P_i, C_t),\ t = 1, \ldots, p\},\ j = 1, \ldots, p$$

If a point P_i belongs to more than one subset M_j it is assigned to the set with least value j.

Step 3: Determine the hyperrectangles $R(M_j)$, $j = 1, ..., p$. Calculate

$$L_\alpha = \text{Max}\{L(M_1), ..., L(M_p)\}.$$

When an algorithm in this class is used for the solution of **(pC)**, Step 3 must consequently be modified to evaluate F_α instead of L_α. The algorithms in this class are different only in the way the seed points are generated (Step 1). The assignment rule given in Step 2, that we will denote by *AR* can be modified if a point P_i belongs to more than one set M_j, for instance assigning it arbitrarily to one M_j containing P_i, but then the value L_α doesn't change. Step 3 is for the evaluation of the objective function of **(RpC)** at a given partition α.

We define any set of hyperrectangles $R_1, ..., R_p$ by two matrices $L_{\min}\, L_{\max} \in M_{\text{mxp}}(\mathbb{R})$ as follows:

$$L_{\min}(k, j) = \text{Minimum}\{x_k(P) : P \in R_j\}$$
$$L_{\max}(k, j) = \text{Maximum}\{x_k(P) : P \in R_j\}$$

Let $L = (L_1, ..., L_p)$, where L_j is the maximum length of a side of R_j, $j = 1, ..., p$. In particular, if $R_j = R(M_j)$ then $L_j = L(M_j)$, $j = 1, ..., p$. An $O(n)$ procedure to evaluate L_α for a given $\alpha = \{M_1, ..., M_p\}$ is:

Procedure RECTANGLE

1. Set $L_{\min}(k, j) = \infty$, $k = 1, ..., m$, $j = 1, ..., p$
   $L_{\max}(k, j) = -\infty$, $k = 1, ..., m$, $j = 1, ..., p$
2. For $j = 1$ To p Do
   While $M_j \neq \emptyset$ Do
   Begin
   Take $P_i \in M_j$
   Set $M_j = M_j - \{P_i\}$
   For $k = 1$ To m Do
   Begin
   If $x_k(P_i) < L_{\min}(k, j)$ then $L_{\min}(k, j) = x_k(P_i)$ Else
   If $x_k(P_i) > L_{\max}(k, j)$ then $L_{\max}(k, j) = x_k(P_i)$
   End
   End

3. Set $L_\alpha = 0$.
   For $j = 1$ To p Do
   For $k = 1$ To m Do
   $L_\alpha = \text{Max}\{L_\alpha, L_{\max}(k, j) - L_{\min}(k, j)\}$

With *RECTANGLE*, the hyperrectangles $R(M_j)$, $j = 1, \ldots, p$, and L_α are determined for a given $\alpha = \{M_1, \ldots, M_p\}$ (Step 3), *i.e.*, the quality of the partition generated by the assignment rule *AR* is evaluated.

We propose a new assignment rule, that we call *NAR* to determine an output partition α from the set of p seed points. The value L_α is calculated at the same time that *NAR* is used to generate α. The new rule starts with the set of p elemental hyperrectangles given by the p seed points $C_1, \ldots, C_p$, which are growing up while there is a non assigned point. In each iteration a non assigned point is assigned to its nearest hyperrectangle, until all the points have been assigned. The distance between a point P_i and an hyperrectangle R_j is here defined as 0 if $P_i \in R_j$ and as the distance between P_i and the farthest point in R_j otherwise, *i.e.*:

$$\bar{d}(P_i, R_j) = \text{Max}\{d_k, k = 1, \ldots, m\}$$

where

$$d_k = \begin{cases} L_{\max}(k, j) - x_k(P_i) & \text{if} \quad x_k(P_i) < L_{\min}(k, j) \\ x_k(P_i) - L_{\min}(k, j) & \text{if} \quad x_k(P_i) > L_{\max}(k, j) \\ 0 & \text{otherwise} \end{cases}$$

Then *NAR* is given by the following procedure:

Procedure NAR

1. Set $L_{\min} = L_{\max} = (C_1, \ldots, C_p)$, and $M_j = \{C_j\}$, $j = 1, \ldots, p$.
   Make unlabelled all the points in *M*.
2. For every point $P_i \in M$ Do
   Begin
   Set $j = 1$
   While $(j \leq p)$ And (P_i unlabelled) Do
   Begin
   Let $d(j) = \bar{d}(P_i, R_j)$
   If $d(j) = 0$ Then $M_j = M_j \cup \{P_i\}$ and label P_i Else $j = j + 1$
   End

If P_i is unlabelled Then
  Begin
    Set $d(e) = \text{Min}\{d(j),\ j = 1, \ldots, p\}$ and $M_e = M_e \cup \{P_i\}$
    For $k = 1$ To m Do
      Begin
        If $x_k(P_i) < L_{\min}(k, e)$ Then $L_{\min}(k, e) = x_k(P_i)$ Else
        If $x_k(P_i) > L_{\max}(k, e)$ Then $L_{\max}(k, e) = x_k(P_i)$
      End
    End
  End

3. Set $L_\alpha = 0$
  For $j = 1$ To p Do
    For $k = 1$ To m Do
      $L_\alpha = \text{Max}\{L_\alpha, L_{\max}(k, j) - L_{\min}(k, j)\}$

As *AR*, *NAR* has $O(pn)$ complexity. In opposition to *AR*, *NAR* determines the objective value of the output partition at the same time the points are assigned to the sets M_j, and so Step 3 is not necessary to evaluate the quality of the generated partition. Thus a new class of algorithms arises with the new rule that can be described as follows:

New class of seed point algorithms for (**RpC**)

Step 1: Generate a set of p seed points $C_1, \ldots, C_p$ in $\mathbb{R}^m$.

Step 2: Execute *NAR*.

Evidently, every algorithm in the first class becomes one algorithm in the new class, using *NAR* once the set of seed points has been generated. Observe that *NAR* depends on the sequence in which the P_i are handled. To compare *NAR* with *AR* we consider the sequence given by increasing index-values of points randomly generated. We will show by computational experiments that for a given set of seed points the quality of the partitions (L_α values) generated by *NAR* is better than the quality obtained when *AR* is used. For this, two algorithms A_1 and A_2 in the first class, and the corresponding algorithms NA_1 and NA_2 in the new class are considered. To describe these algorithms only Step 1 is necessary, which is described respectively by the following two procedures:

Procedure SP1

1. Choose any point P_k in M. Set $C_1 = P_k$, $j = 1$ and calculate $d(P_i) = d(P_i, C_1)$ for each $P_i \in M$.

2. While $j < p$ Do

 Begin

 $C_{j+1} = P_t$ such that $d(P_t) = \text{Max}\{d(P_i),\ i = 1, \ldots, n\}$

 $d(P_i) = \text{Min}\{d(P_i),\ d(P_i, C_{j+1})\}$ for each $P_i \in M$

 End

A_1 is an $O(pn)$ algorithm given by Dyer and Frieze [6] which generates a partition α satisfying $L_\alpha \leq 2L^*$, being L^* the optimal value of **(RpC)**.

Procedure SP2

1. Obtain the smallest hyperrectangle *R(M)* containing the points in *M* and calculate the maximum length *L(M)* of the sides of *R(M)*.
2. Use a dichotomic search to find the least value L in $(0, L(M)]$ for which the following subroutine yields an output set S with $|S| \leq p$,

 RANGE

 Set $S = \emptyset$ and make unlabelled all the points in *M*.

 While there is an unlabelled point in *M* and $|S| \leq p$ Do

 Begin

 Choose an unlabelled point P_t.

 Set $S = S \cup \{P_t\}$

 Label P_t and every unlabelled point P_i.

 such that $d(P_i, P_t) \leq L$.

 End

3. If $|S| = p$ when the search is finished, a set of p seed points has been generated. If $|S| < p$ then add any $p - |S|$ points in $M - S$ to S.

A_2 is a modification of an $O(n^2 \log n)$ algorithm given by Plesnik [16] for points in a network and adapted later for any metric space in [14]. In [14, 16] the search is in the set $D = \{d(P_i, P_k) : P_i, P_k \in M, i < k\}$, which clearly contains L^*, but this is time consuming as it is shown in [15], for instance for $n = 1\,000$, $p = 6$ the run time on an Olivetti M-300 (with an 80386SX processor and 16 Mhz) was 1 534 sec.

In A_2, RANGE is executed each time with the middle point L in an interval $(\underline{L}, \overline{L}]$, initially $\underline{L} = 0$, and $\overline{L} = L(M)$: If $|S| \leq p$ then a new interval is generated taking $\overline{L} = L$, and if $|S| > p$ then a new interval is obtained taking $\underline{L} = L$. The search ends when $|\overline{L} - \underline{L}| < \varepsilon$, for a given $\varepsilon > 0$. The complexity of A_2 is $O(\text{pn} \log(L(M)/\varepsilon))$ which is much less than $O(n^2 \log n)$ for large problems. For A_2, the following properties are verified:

PROPERTY 1: *If $L^* \leq L$ then RANGE generates a set S with $|S| \leq p$.*

Proof: If $L^* \leq L$ then there exists a partition $\alpha = \{M_1, \ldots, M_p\}$ with $L_\alpha \leq L$. Let C_1 denote the first point in S generated by RANGE, then $C_1 \in M_j$ for some index j. If $P_i \in M_j$, as $L(M_j) \leq L_\alpha \leq L$, it follows that $d(P_i, C_1) \leq L$. Then all points in M_j are labelled before a second point C_2 in S will be generated by RANGE. As $d(C_2, C_1) > L$ it follows that $C_2 \notin M_j$. Then $C_2 \in M_h$ for some index $h\,(h \neq j)$, and similarly it can be shown that all the points in M_h are labelled before a third point C_3 in S will be generated by RANGE, and so on. Consequently, when all the points in M are labelled RANGE outputs a set S with $|S| \leq p$. $\square$

PROPERTY 2: *The value $\underline{L}$ obtained at the end of the search in $(0, L(M)]$ is an strict lower bound of L^*.*

Proof: Let $(\underline{L}, \overline{L}]$ be the interval obtained at the end of the search, then $\underline{L}$ is the maximum value found in $(0, L(M)]$ for which RANGE outputs S with $|S| > p$. If $L^* \leq \underline{L}$ then RANGE would output S with $|S| \leq p$ (Property 1), as $|S| > p$ it follows that $\underline{L} < L^*$. $\square$

PROPERTY 3: *A_2 generates a partition α with $L_\alpha < 2(L^* + \varepsilon)$.*

Proof: Let $(\underline{L}, \overline{L}]$ the interval obtained at the end of the search. Then $\overline{L}$ is the least value found in $(0, L(M)]$ for which RANGE outputs a set S with $|S| \leq p$, from which the p seed points $C_1, \ldots, C_p$ are generated. Let $\alpha = \{M_1, \ldots, M_p\}$ be the partition obtained from these seed points: If $P_i, P_h \in M_j$ then $d(P_i, P_h) \leq d(P_i, C_j) + d(P_h, C_j) \leq \overline{L} + \overline{L} = 2\overline{L}$. Consequently

$$L(M_j) = \text{Max}\{d(P_i, P_h) : P_i, P_h \in M_j\} \leq 2\overline{L}$$

for all M_j with two or more points. If M_j is a single point $L(M_j) = 0$. Therefore

$$L_\alpha = \text{Max}\{L(M_1), \ldots, L(M_p)\} \leq 2\overline{L}.$$

Since $\underline{L} < L^*$ and $\overline{L} - \underline{L} < \varepsilon$ then $L_\alpha \leq 2\overline{L} < 2(L^* + \varepsilon)$. $\square$

In [14, 16] the search is finished when the set D reduces to one value L_0 which is a lower bound of L^* and it is verified that $L_\alpha \leq 2L^*$.

3. ALGORITHMS BASED ON PARTITIONS

Most of other proposed heuristic algorithms for **(pC)** start with a partition α_0 and generate a sequence of partitions $\alpha_1, \ldots, \alpha_s$ such that

$F_{\alpha_0} > F_{\alpha_1} > \ldots > F_{\alpha_s}$, until no improvement of F_α for the last generated partition α is obtained. When they are applied to **(RpC)**, the starting partition α_0 can be obtained using any of the algorithms seen in section 2. All of these algorithms can be seen as being of improvement type and the most common are those of location-allocation and exchange types [1, 2, 7, 14]. A different type of algorithm based on partitions is an $O(n^2 \log n)$ algorithm given by Pelegrin [13, 14] for weighted p-center problems, which generates the same value in the set D as the Plesnik algorithm, but a different output partition, when it is used for **(pC)**. A modification of this algorithm for **(RpC)** is as follows:

Algorithm A_3

Step 1: Start with a given interval $(\underline{L}, \overline{L}]$ containing L^*.

Step 2: Use dichotomic search to find the least value in $(\underline{L}, \overline{L}]$ for which the following subroutine yields an output partition α with $|\alpha| \leq p$

PARTITION

Make unlabelled all points in M and $\alpha = \{\emptyset\}$.

While there is an unlabelled point in M Do

Begin

Choose an unlabelled point P_t in M.

Set $M_t = \{P_t\} \cup \{P_i : P_i$ is unlabelled and $d(P_i, P_t) \leq L\}$

Label all points in M_t

Set $\alpha = \alpha \cup \{M_t\}$.

End

Observe that algorithm A_3 is similar to algorithm A_2 changing $(0, L(M)]$ to $(\underline{L}, \overline{L}]$ and *RANGE* to *PARTITION*. *RANGE* generates a set of seed points while *PARTITION* generates a partition each time they are executed. In [13, 14] the search is in the set D, which is time consuming as it is shown in [15].

For the problem **(RpC)**, we propose a new algorithm, that we call NA_3, which is the same as A_3, but changing the subroutine *PARTITION* to the following one:

NEW PARTITION

Make unlabelled all points in M, $\alpha = \{\emptyset\}$, $L_\alpha = 0$.

While there is an unlabelled point in M Do

Begin

Choose an unlabelled point P_t in M.

Set $M_t = \{P_t\}$ and $L(M_t) = 0$.
For every unlabelled point P_i Do
Begin
Evaluate $L(M_t \cup \{P_i\})$
If $L(M_t \cup \{P_i\}) \leq L$ then
label P_i and set $M_t = M_t \cup \{P_i\}$
End
Set $\alpha = \alpha \cup \{M_t\}$, $L_\alpha = \text{Max}\{L_\alpha, L(M_t)\}$
End

PROPERTY 4: *If NEW PARTITION generates α with $|\alpha| \leq p$, for a given L, then $L_\alpha \leq L$.*

Proof: For each subset M_t in α it is verified that $L(M_t) \leq L$. Therefore $L_\alpha = \text{Max}\{L(M_t) : M_t \in \alpha\} \leq L$. □

PARTITION and *NEW PARTITION* are executed each time with the middle point L in an interval $(\underline{L}, \overline{L}]$: If $|\alpha| \leq p$ then a new interval is generated taking $\overline{L} = L$, and if $|\alpha| > p$ then the new interval is obtained taking $\underline{L} = L$. The search ends when $|\overline{L} - \underline{L}| < \varepsilon$ for a given $\varepsilon > 0$. Both algorithms A_3 and NA_3 are $O(pn \log((\overline{L} - \underline{L})/\varepsilon))$ being $(\overline{L}, \underline{L}]$ the starting interval. As starting interval $(\overline{L}, \underline{L}]$ can be used $(0, L(M)]$, or the one taking $\underline{L}$ as the lower bound of L^* obtained by Step 1 of A_2 and NA_2 (*Procedure SP2*), and $\overline{L}$ as L_{α_0} for any partition α_0. If no partition α with $|\alpha| \leq p$ is founded, then no improvement is obtained. Otherwise, the initial partition α_0, for which $L_{\alpha_0} = \overline{L}$, is improved by *NEW PARTITION* since from property 4 the output partition satisfies $L_\alpha \leq L < L_{\alpha_0}$. If $|\alpha| < p$, any $p - |\alpha|$ points $P_{i_1}, \ldots, P_{i_{p-|\alpha|}}$ are eliminated from α to make $\alpha = \alpha \cup \{P_{i_1}\} \cup \ldots \cup \{P_{i_{p-|\alpha|}}\}$, then $|\alpha| = p$. *PARTITION* always generates α verifying $|\alpha| \leq p$ if $L^* \leq L$, see [13, 14]. On the contrary, *NEW PARTITION* can generate α with $|\alpha| > p$ if $L^* \leq L$, for instance with the points P_1 to P_6 in $\mathbb{R}$ of coordinates 2, 3, 4, 5, 1, 0 respectively and $p = 2$, the partition generated for $L = L^* = 2$ satisfies $|\alpha| = 3$.

Finally, we consider the well known Location-Allocation algorithm, which can be described for the problem **(RpC)** as follows:

Algorithm A_4

Step 1: Execute RECTANGLE with the starting partition $\alpha_0 = \{M_1^0, \ldots, M_p^0\}$ to obtain the hyperrectangle $R(M_1^0), \ldots, R(M_p^0)$ and L_{α_0}. Calculate the center C_j^0 of $R(M_j^0)$, $j = 1, \ldots, p$. Set $s = 1$.

Step 2: Use *AR* with the points $C_1^{s-1}, \ldots, C_p^{s-1}$ to obtain a new partition $\alpha_s = \{M_1^s, \ldots, M_p^s\}$. Calculate $R(M_1^s), \ldots, R(M_p^s)$ and L_{α_s}. If $L_{\alpha_s} \geq L_{\alpha_{s-1}}$, output α_{s-1} and STOP. Otherwise, calculate the centers C_j^s of $R(M_j^s)$, $j = 1, \ldots, p$, set $s = s + 1$ and repeat step 2.

This algorithm can also be modified using *NAR* instead of *AR* in Step 2. In such case, a new algorithm is obtained, that we call NA_4. The main advantage of using *NAR* is that this subroutine also determines $R(M_1^s), \ldots, R(M_p^s)$ and L_{α_s} for α_s. When *AR* is used, it is required to use *RECTANGLE* in addition to *AR*, in each iteration. Observe that A_4 and NA_4 reduce to Step 2 if an starting set of centers $C_1^0, \ldots, C_p^0$ is given.

TABLE 0.
Test Problems.

m	n	p	Number of problems
2	$500t$, $t = 1, \ldots, 10$	$2t$, $t = 1, \ldots, 10$	100
3	$500t$, $t = 1, \ldots, 10$	$2t$, $t = 1, \ldots, 10$	100
5	$500t$, $t = 1, \ldots, 10$	$2t$, $t = 1, \ldots, 10$	100
10	$500t$, $t = 1, \ldots, 10$	$2t$, $t = 1, \ldots, 10$	100

4. COMPUTATIONAL EXPERIMENTS

All the algorithms considered in sections 2 and 3 were implemented on an PC compatible with a microprocessor Intel 80486-DX, a math coprocessor 80487 and 50 Mhz. The language used was Turbo Pascal V 6.0. A sample of test problems was obtained by random generation of points in $\mathbb{R}^m$ with integer coordinates in [0, 100] and values of m, n and p given in Table 0. Each algorithm was run for each test problem, realizing four experiments (Tables for these experiments are shown in the appendix).

In experiment 1, we compared algorithms A_1 and NA_1. For each problem, run times in seconds were obtained for the generation of seed points (SP_1), generation of the output partitions using *AR* and *NAR*, and evaluation by *RECTANGLE* of the objective function L_α for α generated by *AR* (*RECT*). Table 1 summarizes the results for each of the p and m values, each row shows average results of 10 test problems, corresponding to $n = 500\,\mathrm{t}$, $\mathrm{t} = 1, \ldots, 10$. The first four columns show average run times of *SP1*, *AR*, *RECTANGLE* and *NAR*, the following two columns show average objective values of the output partitions generated by A_1 and NA_1 and the last three columns show the number of times each algorithm generated the best partition. Observe that the average run times for algorithm A_1 are given by the sum of columns SP_1, *AR* and *RECT* and the average run times for algorithm *NA*$_1$ are given by the sum of columns SP_1 and *NAR*.

In experiment 2, we compared algorithms A_2 and *NA*$_2$ in a similar way to that in experiment 1. Table 2 summarizes the results, showing also the average of the lower bound $\underline{L}$ obtained by the procedure SP_2 (column Av. L.B.).

In experiment 3, we compared A_3 and *NA*$_3$ Run times in seconds were obtained for these algorithms and for the evaluation by *RECTANGLE* of the objective function at each output partition generated by A_3. Table 3 summarizes the results for each of the p and m values, each row shows average results of 10 test problems, corresponding to $n = 500\,\mathrm{t}$, $\mathrm{t} = 1, ..., 10$. The first three columns show average run times of A_3, *RECTANGLE* and *NA*$_3$ the following two columns show average objective values of output partitions generated by A_3 and *NA*$_3$, and the last three columns show the number of times each algorithm generated the best partition.

Finally, in experiment 4, we compared A_4 and *NA*$_4$. Table 4 summarizes the results, showing average run times of these algorithms, average objective values, and the number of times each algorithm generated the best partition for each of the 10 test problems corresponding to each row.

Average run times for *NA*$_1$ and *NA*$_2$ in each dimension were not more than 2.95 sec. higher than for A_1 and A_2 respectively. However the quality of the output partitions was quite notically better for *NA*$_1$ and *NA*$_2$ than for A_1 and A_2 respectively. *NA*$_1$ was superior to A_1 in 338 out of the 400 test problems, and *NA*$_2$ was superior to A_2 in 330 out of the 400 test problems. These results show that the assignment rule NAR seems to be superior to *AR* for the seed point algorithms when they are used for the solution of $(\mathbf{RpC})$.

Average run times for *NA*$_3$ were higher than for A_3, but the quality of the output partitions was superior for *NA*$_3$, this was the best in 291 out of the 400 test problems. Average run times for *NA*$_4$ were less than for A_4, and the quality of the partitions generated by *NA*$_4$ was superior in 233 out of the 400 test problems. Observe also that the average run time of *RECTANGLE* (column RECT) was less than 0.2 sec. in each dimension.

Table 5 shows the number of times each algorithm generated the best partition for each of the 10 test problems corresponding to each of the m and p values. Observe that algorithms A_1, A_2 and A_3 never generated the best partition for $p \geq 4$ and $m = 2, 3, 5$, except A_2 which twice generated the best partition (for $m = 2$, $p = 12$ and $m = 3$, $p = 8$). Algorithms A_1, A_2, A_3 and A_4 never generated the best partition for $p \geq 8$ and $m = 10$. The new algorithms generated the best partitions most of times, NA_3 in 211 times, NA_4 in 150 times, NA_1 in 188 times and NA_2 in 118 times out of the 400 test problems. We conclude that the proposed new algorithms for **(RpC)** improve the related existing algorithms for **(pC)** when they are used for **(RpC)**.

REFERENCES

1. L. Cooper, N-dimensional Location Models: An application to Cluster Analysis, *J. Regional Science*, 1973, *13*, pp. 41-54.
2. Z. Drezner, The p-center problem, Heuristics and Optimal algorithms, *J. Oper. Res. Society*, 1984, *35*, pp. 741-748.
3. Z. Drezner, On the complexity of the exchange algorithm for minimax optimization problems, *Math. Programming*, 1987, *38*, pp. 219-222.
4. Z. Drezner, G. O. Wesolowsky, Single facility L_p distance minimax location, *SIAM J. of Algebraic and Discrete Methods,* 1980, *1*, pp. 315-321.
5. F. Plastria, Solving general continuous single facility location problems by cutting planes, *EJOR,* 1987, *29*, pp. 98-110.
6. M. E. Dyer, A. M. Frieze, A Simple Heuristic for the p-Center Problem, *Oper. Res. Letters*, 1985, *3*, pp. 285-288.
7. H. A. Eiselt, G. Charlesworth, A note on p-center problems in the plane, *Transportation Science*, 1986, *20*, pp. 130-133.
8. J. Elzinga, D. W. Hearn, Geometrical solutions for some minimax location problems, *Transportation Science*, 1972, *6*, pp. 379-394.
9. J. Elzinga, D. W. Hearn, The Minimum Covering Sphere Problem, *Management Science*, 1972, *19*, pp. 96-104.
10. R. L. Francis, J. A. White, *Facility Layout and Location: An analytical approach,* Prentice-Hall, 1974.
11. W. L. Hsu, G. L. Nemhauser, Easy and Hard bottleneck location problems, *Discrete Appl. Math.,* 1979, *1*, pp. 209-216.
12. N. Meggido, K. J. Supowit, On the complexity of some common geometric location problems, *SIAM J. Computing* 1984, *18*, pp. 182-196.
13. B. Pelegrin, The p-center prolem in $\mathbb{R}^n$ with weighted Tchebycheff norms, *JORBEL,* 1991 *a*, *31*, pp. 49-62.
14. B. Pelegrin, Heuristic methods for the p-center problem, *RAIRO*, 1991 *b*, *25*, pp. 65-72.
15. B. Pelegrin, L. Canovas, *A computational study of some heuristic algorithms for the p-center problem with weighted Tchebycheff norms,* 1991 *c*, Presented at OR33 in Exeter, September 1991. United Kingdom.
16. J. Plesnik, A heuristic for the p-center problem in graphs, *Discrete Applied Mathematics*, *17*, pp. 263-268.

APPENDIX

TABLE 1.
Comparison of A1 and NA1.

PROBLEMS		AVERAGE RUN TIME (Sec.)				AV. OBJ. VAL		N. of times g.b.p		
p	*m*	SP1	AR	RECT	NAR	A1	NA1	A1	NA1	Eq.
2	2	0.08	0.11	0.02	0.11	99.0	97.8	0	7	3
	3	0.13	0.12	0.05	0.16	99.0	98.7	0	3	7
	5	0.18	0.18	0.08	0.25	99.0	98.9	0	1	9
	10	0.36	0.35	0.15	0.50	99.0	99.0	0	0	10
4	2	0.21	0.19	0.03	0.23	76.3	61.3	0	10	0
	3	0.29	0.26	0.04	0.20	99.0	96.7	0	9	1
	5	0.43	0.38	0.06	0.31	99.0	98.5	0	5	5
	10	0.74	0.71	0.16	0.63	99.0	99.0	0	0	10
6	2	0.33	0.28	0.03	0.30	67.4	54.4	0	10	0
	3	0.45	0.36	0.06	0.33	99.0	93.4	0	10	0
	5	0.64	0.57	0.09	0.41	99.0	96.7	0	10	0
	10	1.14	1.07	0.16	0.80	99.0	98.5	0	4	6
8	2	0.47	0.37	0.04	0.40	60.1	48.9	0	10	0
	3	0.60	0.50	0.05	0.54	80.8	69.4	1	9	0
	5	0.86	0.78	0.07	0.54	99.0	65.3	0	10	0
	10	1.55	1.43	0.15	1.00	99.0	98.1	0	6	4
10	2	0.58	0.48	0.03	0.50	48.7	40.6	0	10	0
	3	0.76	0.64	0.04	0.64	79.7	60.3	0	10	0
	5	1.09	0.95	0.09	0.66	98.7	94.4	0	10	0
	10	1.91	1.78	0.16	1.30	99.0	97.2	0	10	0
12	2	0.70	0.57	0.03	0.56	46.9	39.2	0	10	0
	3	0.92	0.77	0.03	0.71	76.1	60.8	0	10	0
	5	1.31	1.16	0.06	0.80	99.0	93.6	0	10	0
	10	2.31	2.13	0.15	1.59	99.0	96.6	0	10	0
14	2	0.82	0.66	0.03	0.66	42.2	35.4	0	10	0
	3	1.06	0.89	0.06	0.80	72.5	58.0	1	9	0
	5	1.53	1.34	0.09	1.00	98.8	91.8	0	10	0
	10	2.68	2.49	0.16	1.87	99.0	95.9	0	10	0
16	2	0.94	0.75	0.03	0.76	39.8	34.7	0	10	0
	3	1.21	1.02	0.06	0.94	70.2	56.6	0	10	0
	5	1.75	1.55	0.07	1.20	98.7	90.0	0	10	0
	10	3.08	2.85	0.16	2.23	99.0	95.5	0	10	0
18	2	1.08	0.84	0.03	0.88	36.9	31.9	0	9	1
	3	1.39	1.15	0.04	1.04	66.7	54.1	0	10	0
	5	1.98	1.73	0.08	1.43	99.0	89.2	0	10	0
	10	3.47	3.19	0.17	2.58	99.0	94.7	0	10	0
20	2	1.18	0.94	0.04	0.98	35.0	29.4	1	9	0
	3	1.54	1.27	0.04	1.21	64.9	54.7	0	10	0
	5	2.19	1.94	0.07	1.68	98.9	87.7	0	10	0
	10	3.84	3.57	0.16	2.94	99.0	94.3	0	10	0

TABLE 2.
Comparison of A2 and NA2.

PROBLEMS		AVERAGE RUN TIME (Sec.)				AV. OBJ. VAL.		AV.	N. of times g.b.p		
p	m	SP2	AR	RECT	NAR	A2	NA2	L.B.	A2	NA2	Eq.
2	2	0.49	0.08	0.04	0.11	98.90	98.10	71.62	0	6	4
	3	0.64	0.13	0.04	0.15	99.00	98.70	79.28	0	3	7
	5	0.99	0.20	0.09	0.25	99.00	98.90	87.24	0	1	9
	10	2.02	0.37	0.15	0.49	99.00	99.00	93.35	0	0	10
4	2	0.78	0.19	0.03	0.22	71.60	61.20	43.70	0	10	0
	3	0.80	0.26	0.05	0.20	99.00	96.80	68.06	0	10	0
	5	1.17	0.39	0.08	0.29	99.00	98.50	81.21	0	4	6
	10	2.42	0.73	0.16	0.60	99.00	98.90	88.87	0	1	9
6	2	0.95	0.28	0.04	0.29	60.50	54.00	33.41	1	8	1
	3	0.94	0.39	0.04	0.33	98.80	93.60	59.48	0	10	0
	5	1.33	0.59	0.09	0.40	99.00	96.80	75.80	0	10	0
	10	2.70	1.09	0.15	0.80	99.00	98.40	84.38	0	4	6
8	2	1.01	0.38	0.03	0.37	55.70	51.20	30.16	0	9	1
	3	1.66	0.51	0.06	0.50	85.20	81.20	47.10	4	6	0
	5	1.56	0.77	0.08	0.50	99.00	95.60	72.16	0	10	0
	10	3.16	1.45	0.15	1.04	99.00	97.80	82.68	0	8	2
10	2	1.24	0.47	0.04	0.47	49.00	44.80	26.92	1	9	0
	3	1.85	0.64	0.06	0.62	75.70	65.70	41.30	1	9	0
	5	1.54	0.97	0.08	0.63	99.00	94.00	66.59	0	10	0
	10	3.44	1.82	0.15	1.27	99.00	97.20	79.66	0	9	1
12	2	1.53	0.56	0.04	0.58	44.00	39.10	23.67	1	9	0
	3	2.10	0.77	0.05	0.77	70.40	60.30	38.36	1	9	0
	5	1.86	1.16	0.09	0.82	99.00	93.50	66.28	0	10	0
	10	3.88	2.17	0.16	1.59	99.00	96.70	78.89	0	10	0
14	2	1.88	0.66	0.04	0.66	41.10	35.40	22.20	0	10	0
	3	2.23	0.90	0.04	0.83	68.80	59.80	36.82	1	9	0
	5	1.86	1.36	0.09	0.94	98.60	91.60	64.74	0	10	0
	10	4.15	2.54	0.16	1.89	99.00	96.50	76.03	0	10	0
16	2	2.03	0.75	0.03	0.76	36.60	34.00	19.34	0	10	0
	3	2.33	1.02	0.06	0.95	66.50	57.60	35.27	1	9	0
	5	2.12	1.56	0.08	1.13	98.70	90.80	61.26	0	10	0
	10	4.52	2.89	0.16	2.21	99.00	95.60	75.26	0	10	0
18	2	2.23	0.84	0.04	0.86	35.40	32.50	18.34	1	8	1
	3	2.54	1.15	0.05	1.05	63.50	55.30	33.34	0	10	0
	5	2.27	1.77	0.07	1.37	98.60	89.30	60.41	0	10	0
	10	4.99	3.25	0.17	2.56	99.00	94.90	75.18	0	10	0
20	2	2.30	0.96	0.04	0.97	32.90	29.60	17.40	1	9	0
	3	2.73	1.29	0.06	1.19	60.60	52.70	32.48	0	10	0
	5	2.39	1.95	0.07	1.59	98.60	87.20	57.62	0	10	0
	10	5.46	3.62	0.18	2.92	99.00	94.90	73.48	0	10	0

TABLE 3.
Comparison of A3 and NA3.

PROBLEMS		AVERAGE RUN TIME (Sec.)			AV. OBJ. VAL.		N. of times g.b.p.		
p	*m*	A3	RECT	NA3	A3	NA3	A3	NA3	Eq.
2	2	0.54	0.01	0.84	98.70	98.20	1	7	2
	3	0.69	0.03	1.20	99.00	98.90	0	1	9
	5	1.05	0.04	1.97	99.00	98.90	0	1	9
	10	2.08	0.41	4.53	99.00	99.00	0	0	10
4	2	0.85	0.02	0.95	84.90	94.70	9	1	0
	3	0.89	0.02	1.45	99.00	97.10	0	10	0
	5	1.24	0.03	2.58	99.00	97.80	0	10	0
	10	2.52	0.41	6.06	99.00	98.70	0	3	7
6	2	1.09	0.01	1.58	66.90	69.20	5	4	1
	3	1.02	0.03	1.67	98.70	94.00	0	10	0
	5	1.41	0.04	3.05	99.00	96.90	0	10	0
	10	2.79	0.42	7.52	99.00	98.00	0	8	2
8	2	1.16	0.02	2.53	60.60	48.90	0	10	0
	3	1.81	0.03	1.84	92.60	90.60	2	8	0
	5	1.64	0.03	3.51	99.00	95.00	0	10	0
	10	3.33	0.41	8.82	99.00	97.10	0	10	0
10	2	1.39	0.02	2.70	54.60	47.50	0	9	1
	3	2.04	0.01	2.12	83.80	85.20	4	3	3
	5	1.61	0.04	3.94	99.00	93.30	0	10	0
	10	3.63	0.42	10.1	99.00	96.10	0	10	0
12	2	1.83	0.01	2.81	48.00	45.60	1	7	2
	3	2.32	0.02	2.49	77.30	79.40	6	4	0
	5	2.01	0.03	4.37	99.00	91.90	0	10	0
	10	4.02	0.40	11.4	99.00	95.20	0	10	0
14	2	2.09	0.03	3.17	45.20	41.80	0	9	1
	3	2.47	0.02	3.60	74.40	68.80	3	7	0
	5	2.09	0.04	4.78	98.90	90.00	0	0	10
	10	4.24	0.39	12.6	99.00	94.60	0	10	0
16	2	2.30	0.01	3.52	39.60	37.60	2	8	0
	3	2.58	0.02	4.34	71.40	62.30	1	9	0
	5	2.46	0.04	5.20	98.80	88.70	0	10	0
	10	5.03	0.42	13.9	99.00	94.20	0	10	0
18	2	2.52	0.02	3.62	38.20	35.40	1	3	6
	3	2.76	0.02	6.72	67.20	51.60	0	10	0
	5	2.66	0.04	5.57	99.00	87.50	0	10	0
	10	5.10	0.42	15.0	99.00	93.40	0	10	0
20	2	2.63	0.02	4.33	35.80	31.20	1	9	0
	3	3.14	0.03	7.03	65.00	49.30	0	10	0
	5	2.79	0.05	5.97	99.00	86.00	0	10	0
	10	5.87	0.41	16.4	99.00	92.50	0	10	0

TABLE 4.
Comparison of A4 and NA4.

PROBLEMS		AV. RUN TIME (Sec)		AV. OBJ. VAL.		N. of times g.b.p.		
p	*m*	A4	NA4	A4	NA4	A4	NA4	Eq.
2	2	0.32	0.21	98.50	98.10	1	6	3
	3	0.43	0.27	99.00	98.80	0	2	8
	5	0.67	0.44	99.00	98.90	0	1	9
	10	1.27	0.85	99.00	99.00	0	0	10
4	2	2.85	0.53	65.90	61.20	6	4	0
	3	0.78	0.39	98.90	96.00	0	10	0
	5	1.17	0.59	99.00	98.20	0	7	3
	10	2.23	1.06	99.00	98.80	0	2	8
6	2	2.45	0.78	54.60	51.30	5	2	3
	3	1.38	0.63	97.50	92.90	0	10	0
	5	1.81	0.87	98.60	96.90	0	10	0
	10	3.17	1.44	99.00	98.40	0	4	6
8	2	2.86	0.88	49.10	50.40	6	3	1
	3	2.26	1.29	95.10	73.70	0	10	0
	5	2.39	1.05	98.10	95.40	1	8	1
	10	4.19	1.90	98.80	97.40	0	9	1
10	2	5.88	1.08	42.30	45.70	6	3	1
	3	9.56	1.93	61.10	63.60	7	3	0
	5	4.36	1.14	96.40	93.80	0	9	1
	10	5.10	2.59	98.90	96.50	0	10	0
12	2	7.01	1.69	38.80	40.20	7	2	1
	3	10.72	2.02	53.20	61.50	9	1	0
	5	5.03	1.60	95.60	92.60	1	9	0
	10	6.12	2.79	98.90	96.20	0	10	0
14	2	7.44	1.63	37.20	37.80	6	2	2
	3	8.78	2.30	52.20	56.10	7	2	1
	5	5.85	2.10	95.50	90.40	1	8	1
	10	6.95	3.77	99.00	95.80	0	10	0
16	2	7.53	1.81	34.50	35.70	5	4	1
	3	10.51	2.20	50.50	56.45	10	0	0
	5	7.62	2.40	94.40	88.90	0	9	1
	10	7.98	3.81	98.80	95.40	0	10	0
18	2	6.14	2.67	33.30	32.40	2	5	3
	3	9.63	2.93	50.60	54.00	8	2	0
	5	7.92	3.29	94.10	88.60	0	10	0
	10	10.14	5.72	98.40	94.50	0	10	0
20	2	8.93	3.06	32.30	30.30	2	6	2
	3	10.46	3.19	50.20	51.60	5	0	5
	5	10.60	3.87	92.60	87.20	0	10	0
	10	11.12	5.89	98.50	94.50	0	10	0

TABLE 5.
Global results.

Number of Times each algorithm generated the best partition									
m	*p*	A1	A2	A3	A4	NA1	NA2	NA3	NA4
2	2	2	2	2	3	9	8	7	8
	4	0	0	0	6	4	1	0	1
	6	0	0	0	7	1	1	2	3
	8	0	0	0	3	5	2	7	3
	10	0	0	0	4	6	1	0	0
	12	0	1	0	6	2	1	0	2
	14	0	0	0	4	5	2	0	0
	16	0	0	0	2	5	4	2	3
	18	0	0	0	1	4	3	1	3
	20	0	0	0	2	6	5	0	4
3	2	6	6	6	6	9	9	7	8
	4	0	0	0	0	6	4	4	8
	6	0	0	0	0	3	3	3	5
	8	0	1	0	0	6	1	0	2
	10	0	0	0	5	4	1	0	1
	12	0	0	0	9	0	1	0	0
	14	0	0	0	7	1	0	1	3
	16	0	0	0	8	0	2	1	0
	18	0	0	0	3	1	0	7	2
	20	0	0	0	3	1	2	8	2
5	2	9	9	9	9	10	10	10	10
	4	0	0	0	0	3	3	10	6
	6	0	0	0	0	5	5	4	5
	8	0	0	0	0	3	3	7	5
	10	0	0	0	0	2	5	6	5
	12	0	0	0	1	1	1	7	3
	14	0	0	0	1	2	2	6	4
	16	0	0	0	0	3	2	6	5
	18	0	0	0	0	2	1	7	4
	20	0	0	0	0	2	4	4	3
10	2	10	10	10	10	10	10	10	10
	4	6	6	6	6	6	7	9	8
	6	2	2	2	2	3	5	8	5
	8	0	0	0	0	1	1	7	5
	10	0	0	0	0	1	1	10	6
	12	0	0	0	0	4	1	10	2
	14	0	0	0	0	1	0	10	1
	16	0	0	0	0	2	2	10	1
	18	0	0	0	0	2	1	10	3
	20	0	0	0	0	1	0	10	1
TOTAL		35	37	35	108	141	118	211	150