A branch-and-price approach to p-median location problems

Edson L. F. Senne*
FAX: +55 (12) 3123-2868
elfsenne@feg.unesp.br
FEG/UNESP – São Paulo State University
Engineering School – Mathematics Department
12.500-000 Guaratinguetá, SP - Brazil

Luiz A. N. Lorena
lorena@lac.inpe.br
LAC/INPE – Brazilian Space Research Institute
PO Box 515
12.201-970 São José dos Campos, SP – Brazil

Marcos A. Pereira
marcos@lac.inpe.br
LAC/INPE – Brazilian Space Research Institute
PO Box 515
12.201-970 São José dos Campos, SP – Brazil

* corresponding author
Abstract

This paper describes a branch-and-price algorithm for the $p$-median location problem. The objective is to locate $p$ facilities (medians) such as the sum of the distances from each demand point to its nearest facility is minimized. The traditional column generation process is compared with a stabilized approach that combines the column generation and Lagrangean/surrogate relaxation. The Lagrangean/surrogate multiplier modifies the reduced cost criterion, providing the selection of new productive columns at the search tree. Computational experiments are conducted considering especially difficult instances to the traditional column generation and also with some large-scale instances.

Keywords: $p$-median, column generation, Lagrangean/surrogate relaxation, branch-and-price.
1 Introduction

This paper describes a branch-and-price algorithm for the $p$-median location problem. The search for $p$-median nodes on a network is a classical location problem. The objective is to locate $p$ facilities (medians) such that the sum of the distances from each demand point to its nearest facility is minimized. The problem is well known to be NP-hard and several heuristics have been developed for $p$-median problems.

The combined use of Lagrangean/surrogate relaxation and subgradient optimization in a primal-dual viewpoint was found to be a good solution approach to the problem [19]. The Lagrangean/surrogate generalizes the standard Lagrangean relaxation using the local surrogate information of constraints relaxed in Lagrangean relaxation, in order to accelerate subgradient-like methods. A local search is conducted at some initial iteration of the subgradient algorithm, correcting wrong step sizes. The gain in computational times can be substantial for large-scale problems [17, 19].

Column generation is a powerful tool for solving large-scale linear programming problems. Such linear programming problems may arise when the columns in the problem are not known in advance and a complete enumeration of all columns is not an option, or the problem is rewritten using Dantzig-Wolfe decomposition (the columns correspond to all extreme points of a certain constraint set) [5]. Column generation is explored in several applications, such as the well-known cutting-stock problem, vehicle routing and crew scheduling [6, 7, 8, 12, 13, 22, 23, 24]. In classical implementations of column generation, the algorithm iterates between a restricted
master problem and a column generation subproblem. Solving the master problem yields a certain dual solution, which is used in the subproblem to determine whether there is any column that might be an incoming column.

In many cases a straightforward application of column generation may result in slow convergence [18]. Senne and Lorena [20] recently presented a stabilized algorithm to \( p \)-median problems (see also [15]). The Lagrangean/surrogate relaxation performs as an acceleration process to column generation, generating new productive sets of columns. The known stability problems of column generation are reduced, mainly for large-scale problems. Other attempts to stabilize the dual appeared before, like the Boxstep method [16], the Bundle methods and the Analytic Center Cutting Plane method [9]. Neame [18] describes these and other recent alternative methods to stabilize the dual [10].

The branch-and-price [1] method was initially proposed to solve large-scale combinatorial optimization problems. It is implemented as a search tree algorithm employing column generation at each search node. We find applications of this method in generalized assignment, crew scheduling [1] and capacitated \( p \)-median problem [3], among others.

In this work, some aspects, like the branching rule and the tree search, are readily determined considering the uncapacitated \( p \)-median problem as a clustering problem. The Lagrangean/surrogate multiplier modifies the reduced cost criterion, providing the selection of new productive columns at the root node and also at the search tree
nodes. Computational experiments are conducted comparing the Lagrangean/surrogate and Lagrangean relaxations in a branch-and-price approach, considering especially difficult instances to the traditional column generation and also with some large-scale instances.

The paper is organized as follows. Section 2 summarizes the stabilized column generation approach to $p$-median presented in [20]. Section 3 presents the relevant aspects considered in our branch-and-price implementation. Section 4 presents the algorithms and Section 5 gives computational results.

## 2 A stabilized column generation for $p$-median problems

The $p$-median problem considered in this paper can be formulated as the following binary integer-programming problem:

$$\text{(Pmed):} \quad v(Pmed) = \min \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} x_{ij}$$

subject to

$$\sum_{j=1}^{n} x_{ij} = 1 \quad j \in N$$ \hspace{1cm} (1)

$$\sum_{i=1}^{n} x_{ni} = p$$ \hspace{1cm} (2)

$$x_{ij} \leq x_{ii} \quad i, j \in N$$ \hspace{1cm} (3)

$$x_{ij} \in \{0,1\} \quad i, j \in N$$ \hspace{1cm} (4)
where:

$[d_{ij}]_{n \times n}$ is a symmetric cost (distance) matrix, with $d_{ii} = 0, \forall i \in N$;

$p$ is the number of facilities (medians) to be located;

$n$ is the number of nodes in the network, $N = \{1, ..., n\}$;

$[x_{ij}]_{n \times n}$ is the allocation matrix, with $x_{ij} = 1$ if node $j$ is allocated to median $i$, and $x_{ij} = 0$, otherwise; $x_{ii} = 1$ if node $i$ is a median and $x_{ii} = 0$, otherwise.

Constraints (1) and (3) ensure that each node $j$ is allocated to only one node $i$, which must be a median. Constraint (2) determines the exact number $p$ of medians to be located, and (4) gives the integer conditions.

The problem $(P_{med})$ is a classical formulation explored in many papers. Garfinkel et al. [11] and Swain [21] proposed a set partition formulation to $(P_{med})$, given by:

\[
(\text{SP-Pmed}): \quad v(SP - P_{med}) = \min \sum_{k=1}^{m} c_k x_k
\]

subject to

\[
\sum_{k=1}^{m} A_k x_k = 1 \quad (5)
\]

\[
\sum_{k=1}^{m} x_k = p \quad (6)
\]

\[
x_k \in \{0,1\},
\]

where:
$S = \{S_1, \ldots, S_m\}$, is the set of all subsets of $N$;

$A_k = [a_{ik}]_{n \times 1}$, with $a_{ik} = 1$, if $i \in S_k$, and $a_{ik} = 0$ otherwise;

$[x_k]_{m \times 1}$ indicates if subset $S_k$ is in the solution ($x_k = 1$), or not ($x_k = 0$); and

$$c_k = \min_{j \in S_k} \left( \sum_{i \in S_k} d_{ij} \right).$$

In this formulation (set partitioning problem with cardinality constraint), each subset $S_k$ corresponds to a column $A_k$ in constraint set (5), representing a cluster in which the median node is decided when the cost $c_k$ is calculated, and so the columns of ($SP-Pmed$) implicitly consider the constraint set (3) in ($Pmed$). Constraints (1) and (2) are conserved and respectively updated to (5) and (6).

The Dantzig-Wolfe decomposition process is applied to ($SP-Pmed$) by relaxing the integrality requirements ($x_k \in [0,1]$). A restricted master problem ($LP-Pmed$) is then defined, in the column generation context [1], by dropping columns in the formulation of ($SP-Pmed$).

Senne and Lorena [19] presented the Lagrangean/surrogate relaxation for the $p$-median problem. A general description of the Lagrangean/surrogate relaxation appeared in the work of Narciso and Lorena [17]. For a given $t \in R$ and $\pi \in R^n$, the Lagrangean/surrogate relaxation of problem ($Pmed$) is given by:
(LS\textsubscript{Pmed\textsuperscript{p}}): \quad v(LS_{Pmed\textsuperscript{p}}) = \text{Min} \sum_{i=1}^{n} \sum_{j=1}^{n} (d_{ij} - t\pi_j)x_{ij} + t\sum_{j=1}^{n} \pi_j \\
subject to \quad (2), (3) and (4)

Problem (LS\textsubscript{Pmed\textsuperscript{p}}) is solved considering constraint (2) implicitly and decomposing the problem for index \(i\), obtaining the following \(n\) subproblems:

\[
\text{Min} \sum_{j=1}^{n} (d_{ij} - t\pi_j)x_{ij}
\]

subject to (3) and (4).

Each problem is easily solved letting \(\beta_i = \sum_{j=1}^{n} [\text{Min} \{0, d_{ij} - t\pi_j\}]\), and choosing \(I\) as the index set of the \(p\) smallest \(\beta_i\) (here constraint (2) is implicitly considered).

Then, a solution \(x_{ij}^*\) to problem (LS\textsubscript{Pmed\textsuperscript{p}}) is:

\[
x_{ij}^* = \begin{cases} 
1, & \text{if } i \in I \\
0, & \text{otherwise}
\end{cases}
\]

and for all \(i \neq j\),

\[
x_{ij}^* = \begin{cases} 
1, & \text{if } i \in I \text{ and } d_{ij} - t\pi_j < 0 \\
0, & \text{otherwise}
\end{cases}
\]
The Lagrangean/surrogate solution is given by 
\[ v(LS,Pmed) = \sum_{j=1}^{n} \beta_j x_{ij} + t \sum_{j=1}^{n} \pi_j. \]

The usual Lagrangean relaxation results from \((LS,Pmed)\) if \(t\) is set to 1. For a fixed multiplier \(\pi\), the best value for \(t\) can be found solving approximately a local Lagrangean dual \(\max_{r \in R} v(LS,Pmed)\). A dichotomous search used to find an approximate value for \(t\) is presented in [19].

The Lagrangean/surrogate is integrated to the column generation process transferring the multipliers \(\pi_j (j = 1, ..., n)\) of the restricted master problem \((LP-Pmed)\) to the problem \(\max_{r \in R} v(LS,Pmed)\). The median (and its allocated non-median nodes) with smallest contribution on \(v[\max_{r \in R} v(LS,Pmed)]\) results to be the one selected to produce the incoming column on the subproblem:

\[ (Sub,Pmed): v(Sub,Pmed) = \min_{a \in \mathbb{N}} \left[ \min_{a_{ij} \in \{0,1\}} \left( \sum_{j=1}^{n} (d_{ij} - t \pi_j) a_{ij} \right) \right] \]

Let \(\alpha\) be the dual variable corresponding to constraint (6) and \(j^*\) be the vertex index reaching the overall minimum on \(v(Sub,Pmed)\). The new sets \(S_j\) are \(\{i : a_{ij} = 1\}\) in \((Sub,Pmed)\) and the column \(\frac{A_j}{1}\) is added to \((LP-Pmed)\) if
\[
\left[ \sum_{j=1}^{n} \left( d_{ij} - \pi_j +_{ij} \right) \alpha_{ij} \right] < |\alpha|.
\]
In effect, all the corresponding columns \( \begin{bmatrix} A_j \\ 1 \end{bmatrix} \) satisfying the expression:

\[
\left[ \text{Min} \sum_{j=1}^{n} \left( d_{ij} - \pi_j \right) \alpha_{ij} \right] < |\alpha|, \quad (7)
\]
can be added to the pool of columns. For \( t = 1 \) (Lagrangean case) this is also known as multi-pricing in column generation context [1].

3. The branch-and-price steps

The branch-and-price is detailed in this section, examining separately, the root node, the branching rule, and the search tree and pruning conditions.

3.1 The root node

The following algorithm is used on the root node:

\textbf{Algorithm CG(1)}

(i) Set an initial pool of columns to (LP-Pmed);

(ii) Solve (LP-Pmed) using CPLEX [14] and return the duals prices \( \pi_j, j = 1, \ldots, n \) and \( \alpha \).
(iii) Solve approximately (by a dichotomous search) a local Lagrangean/surrogate dual \( \max_{t \in R} v(LS_tPmed^x) \), returning the corresponding columns of \((Sub_tPmed)\);

(iv) Append to \((SP-Pmed)\) the columns \( \begin{bmatrix} A_j \\ 1 \end{bmatrix} \) satisfying expression (7);

(v) If no columns are found in step (iv) or \( |v(SP - Pmed) - v(LS_tPmed^x)| < 1 \), then stop;

(vi) Perform tests to remove columns and return to (ii).

Assuming \( t = 1 \), the algorithm CG(1) gives the traditional column generation process. In this case, the dichotomous search in step (iii) is not necessary and the usual Lagrangean bound \((LS_1Pmed^x)\) implicitly solves the \((Sub_1Pmed)\) problem. In any case, the bounds \( v(LP-Pmed) \) and \( v(LS_tPmed^x) \) are calculated at each iteration.

The following algorithm is used to set an initial pool of columns to \((LP-Pmed)\):

**Algorithm IC**

Let

\[ Max_{Cols} \]

be the maximum number of columns for the initial pool of columns.

\[ ncols = 0; \]

While (\( ncols < Max_{Cols} \)) do

Let \( M = \{ n_1, ..., n_p \} \subset N \) be a randomly generated set of nodes.
For each $i = 1, ..., p$ do

$$S_i = \{n_j\} \cup \{k \in N \setminus M : d_{kn_i} = \text{Min}(d_{kj}) \}$$

$$c_i = \text{Min}_{j \in S_i} \left( \sum_{k \in S_i} d_{jk} \right)$$

For $j = 1, ..., n$ do

Set $a_{ij} = 1$ if $j \in S_i$

$$a_{ij} = 0$$, otherwise

End_for

Include column \( \begin{bmatrix} A_i \\ 1 \end{bmatrix} \) in the initial pool of columns.

End_For;

ncols = ncols + p;

End_While;

The following algorithm is used in step (vi) of $CG(t)$:

**Algorithm RC**

Let

- $rc_{\text{mean}}$ be the average of the reduced costs for the initial pool of columns (after algorithm $IC$) of ($LP-Pmed$)
- $m$ be the total number of columns in the current ($LP-Pmed$)
- $rc_i$ be the reduced cost of the columns in the current ($LP-Pmed$) ($i = 1, ..., m$)
For $i = 1, \ldots, m$ do

Delete column $i$ from the current $(LP-Pmed)$ if $rc_i > rc_{\text{mean}}$.

End_For;

### 3.2 The branching rule

The branching rule considers the partitioning with identical subsets described in Wolsey [22]. Let $q$ and $r$ be the indices of rows presenting pairs of fractional columns like

\[
\begin{pmatrix}
0 & 1 \\
1 & 1
\end{pmatrix}
\]

in the restricted master final solution. The pair $(q, r)$ is identified as follows.

Assume that $|A|$ denotes the cardinality of a set $A$. Let $X = \{x_1, \ldots, x_m\}$ be the set of fractional decision variables corresponding to the set $S = \{S_1, \ldots, S_m\}$ of columns of $(LP-Pmed)$. Let $QS(i) = \{S_j : i \in S_j, j = 1, \ldots, m\}$ for each row index $i$ ($i = 1, \ldots, n$). Then, $q$ is chosen as the row index such that $|QS(q)| > |QS(i)|$, $\forall i$ ($i = 1, \ldots, n$). Note that, if $|QS(q)| = 1$, then $X$ is a feasible solution of $(LP-Pmed)$.

Let $RS(i) = \{S_j \in QS(q) : i \in S_j, j = 1, \ldots, m\}$ for each row index $i$ ($i = 1, \ldots, n$). Let $T$ be the set of row index $i$ for which the set $RS(i)$ is non-empty, that is, $T = \{i : RS(i) \neq \emptyset, i = 1, \ldots, n\}$. Then, $r$ is chosen as the row index such that $|RS(r)| < |RS(i)|$, $\forall i \in T$. 


Once determined the pair \((q, r)\) of row indices, the following problem is solved on the left branch nodes:

\[
v(\text{SubPmed}) = \min_{j \in N} \left[ \min_{a_{ij} \in \{0,1\}} \sum_{i=1}^{n} (d_{ij} - \tau_{ij})a_{ij} \right]
\]

subject to

\[
a_{qj} = a_{rj}, \quad j = 1, \ldots, n.
\]

And, on the right branch nodes, the following problem is solved:

\[
v(\text{SubPmed}) = \min_{j \in N} \left[ \min_{a_{ij} \in \{0,1\}} \sum_{i=1}^{n} (d_{ij} - \tau_{ij})a_{ij} \right]
\]

subject to

\[
a_{qj} + a_{rj} \leq 1, \quad j = 1, \ldots, n.
\]

Observe that the integer binary problem at each node of the search tree is a combination of several problems such as (8) and (9), depending on the path from the root to the considered node.

### 3.3 The search tree and pruning conditions

The search tree is built in a depth first search and, at each node, the corresponding problems (8) and (9) are solved using CPLEX [14]. A tree node is pruned if the corresponding \(v(LP-Pmed)\) or \(v(LS,Pmed^p)\) are not less than the current best feasible solution.
4 The computational tests

The branch-and-price method described above was coded in C language and executed in a Pentium III 1.13 GHz microcomputer. We compared the computational results of the branch-and-price algorithms, when the variable $t$ is fixed to 1, corresponding to the Lagrangean relaxation case, and for $t$ calculated by dichotomous search, corresponding to the Lagrangean/surrogate relaxation. Table I shows the results for some problems obtained from OR-Library [2]. In this table:

- $n$ is the number of nodes;
- $p$ is the number of medians;
- solution is the calculated optimal solution value;
- Num_Cols is the maximum problem size, in terms of number of columns;
- Tree size is the number of nodes generated;
- Time is the total computational time (in seconds).

Table I – Results for OR-Library instances.

As can be observed from Table I, the branch-and-price approach using the Lagrangean/surrogate relaxation explores smaller trees to obtain new (fewer) columns to the restricted master problem, generally resulting in smaller computational times.
Christofides and Beasley [4] observed that the relation $n/p$ determines the complexity of instances of $p$-median problems solved by the combined use of Lagrangean relaxation and subgradient optimization. In the column generation approach, the bigger the relation $n/p$, the harder the solution process. Table II presents the results of algorithms $CG(1)$ and $CG(t)$ for instances where $n/p \geq 10$.

Table II – Results for OR-Library harder instances.

The results confirm the superiority of Lagrangean/surrogate relaxation in the column generation process.

5 Conclusion

In this paper we present a branch-and-price method for $p$-median location problems. This method uses a column generation process that differs from the traditional because employs Lagrangean/surrogate relaxation. The Lagrangean/surrogate multiplier modifies the reduced cost criterion, providing the selection of more productive columns at each search tree node than the traditional column generation approach. The algorithm proposed is able to find the optimal solution of $p$-median problems exploring smaller search trees, and in shorter computational times.
Acknowledgments: The authors acknowledge Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (processes 302408/88-6, 300837/89-5 and 380758/02-4, respectively) for partial financial research support. The useful suggestions made by two anonymous referees are also greatly acknowledged.

References


Vitae

Edson Luiz França Senne teaches computer science in the Mathematics Department at São Paulo State University, Campus of Guaratinguetá. He received the Doctorate in Applied Computation in 1987 from INPE – Brazilian Space Research Institute. His research interest includes combinatorial optimization applications, decision support systems and geographical information systems. He has published in European Journal of Operational Research, Journal of Intelligent Systems and International Journal of Mathematical Algorithms.

Luiz Antonio Nogueira Lorena is a senior researcher of the Applied Mathematics and Computation Laboratory (LAC) at INPE (Brazilian Space Research Institute), Brazil. He received a Dr. Degree in System Engineering and Computation (1985) from COPPE at Federal University of Rio de Janeiro, Brazil. His main research activities are on combinatorial optimization applications with particular emphasis in heuristics. He has published in European Journal of Operational Research, Journal of the Operational Research Society, Evolutionary Computation, Geoinformatica, IEEE TCAD, International Journal of Industrial Engineering, and International Journal of Mathematical Algorithms.

Marcos Antonio Pereira is a Doctorate student at the Applied Mathematics and Computation Laboratory (LAC) at INPE (Brazilian Space Research Institute), Brazil, under supervision of Dr. Luiz A. N. Lorena. He received a Master Degree in Applied Mathematics from State University of Campinas – UNICAMP (1993). His research
interest includes combinatorial optimization and geographical information systems.

He published in International Journal of Industrial Engineering.
Table I – Computational results for OR-Library instances

<table>
<thead>
<tr>
<th>n</th>
<th>p</th>
<th>solution</th>
<th>Branch-and-price Lagrangean</th>
<th>Branch-and-price Lagrangean/surrogate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Num_Cols</td>
<td>Tree size</td>
</tr>
<tr>
<td>100</td>
<td>20</td>
<td>3034</td>
<td>2172</td>
<td>1</td>
</tr>
<tr>
<td>100</td>
<td>33</td>
<td>1355</td>
<td>1614</td>
<td>1</td>
</tr>
<tr>
<td>200</td>
<td>40</td>
<td>2734</td>
<td>3344</td>
<td>9</td>
</tr>
<tr>
<td>200</td>
<td>67</td>
<td>1255</td>
<td>2046</td>
<td>3</td>
</tr>
<tr>
<td>300</td>
<td>60</td>
<td>2968</td>
<td>4051</td>
<td>15</td>
</tr>
<tr>
<td>300</td>
<td>100</td>
<td>1729</td>
<td>2352</td>
<td>1</td>
</tr>
<tr>
<td>400</td>
<td>80</td>
<td>2845</td>
<td>3669</td>
<td>1</td>
</tr>
<tr>
<td>400</td>
<td>133</td>
<td>1789</td>
<td>2798</td>
<td>1</td>
</tr>
<tr>
<td>500</td>
<td>100</td>
<td>2961</td>
<td>23191</td>
<td>9</td>
</tr>
<tr>
<td>500</td>
<td>167</td>
<td>1828</td>
<td>3237</td>
<td>3</td>
</tr>
<tr>
<td>600</td>
<td>120</td>
<td>3033</td>
<td>28357</td>
<td>25</td>
</tr>
<tr>
<td>600</td>
<td>200</td>
<td>1989</td>
<td>3499</td>
<td>7</td>
</tr>
<tr>
<td>700</td>
<td>140</td>
<td>3013</td>
<td>48579</td>
<td>3</td>
</tr>
<tr>
<td>700</td>
<td>233</td>
<td>1847</td>
<td>3950</td>
<td>5</td>
</tr>
<tr>
<td>800</td>
<td>267</td>
<td>2026</td>
<td>4966</td>
<td>21</td>
</tr>
<tr>
<td>900</td>
<td>300</td>
<td>2106</td>
<td>3900</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Average</strong></td>
<td><strong>8857,8</strong></td>
</tr>
</tbody>
</table>

The columns contains:

- **n** is the number of nodes;
- **p** is the number of medians;
- **solution** is the calculated optimal solution value;
- **Num_Cols** is the maximum problem size, in terms of number of columns;
- **Tree size** is the number of nodes generated;
- **Time** is the total computational time (in seconds).
Table II – Results for OR-Library harder instances

<table>
<thead>
<tr>
<th>n</th>
<th>p</th>
<th>solution</th>
<th>Branch-and-price</th>
<th>Branch-and-price</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Lagrangean</td>
<td>Lagrangean/surrogate</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Num_Cols</td>
<td>Tree size</td>
</tr>
<tr>
<td>100</td>
<td>5</td>
<td>5819</td>
<td>3230</td>
<td>1</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>4093</td>
<td>5510</td>
<td>9</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>4250</td>
<td>3626</td>
<td>31</td>
</tr>
<tr>
<td>200</td>
<td>20</td>
<td>4445</td>
<td>7657</td>
<td>1</td>
</tr>
<tr>
<td>300</td>
<td>30</td>
<td>4374</td>
<td>6686</td>
<td>1</td>
</tr>
<tr>
<td>400</td>
<td>40</td>
<td>4809</td>
<td>54830</td>
<td>39</td>
</tr>
<tr>
<td>500</td>
<td>50</td>
<td>4619</td>
<td>40243</td>
<td>3</td>
</tr>
</tbody>
</table>

The columns contains:

- \( n \) is the number of nodes;
- \( p \) is the number of medians;
- solution is the calculated optimal solution value;
- Num_Cols is the maximum problem size, in terms of number of columns;
- Tree size is the number of nodes generated;
- Time is the total computational time (in seconds).

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Average</strong></td>
<td><strong>17397,4</strong></td>
<td><strong>12,1</strong></td>
<td><strong>5996,0</strong></td>
<td><strong>7802,7</strong></td>
<td><strong>8,1</strong></td>
</tr>
</tbody>
</table>