A Hybrid Data Mining Metaheuristic for the p-Median Problem*

Alexandre Plastino† Erick R. Fonseca‡ Richard Fuchshuber§ Simone de L. Martins¶
Alex A. Freitas‖ Martino Luis** Said Salhi††

Abstract

Metaheuristics represent an important class of techniques to solve, approximately, hard combinatorial optimization problems for which the use of exact methods is impractical. In this work, we propose a hybrid version of the GRASP metaheuristic, which incorporates a data mining process, to solve the p-median problem. We believe that patterns obtained by a data mining technique, from a set of sub-optimal solutions of a combinatorial optimization problem, can be used to guide metaheuristic procedures in the search for better solutions. Traditional GRASP is an iterative metaheuristic which returns the best solution reached over all iterations. In the hybrid GRASP proposal, after executing a significant number of iterations, the data mining process extracts patterns from an elite set of sub-optimal solutions for the p-median problem. These patterns present characteristics of near optimal solutions and can be used to guide the following GRASP iterations in the search through the combinatorial solution space. Computational experiments, comparing traditional GRASP and different data mining hybrid proposals for the p-median problem, showed that employing patterns mined from an elite set of sub-optimal solutions made the hybrid GRASP find better results. Besides, the conducted experiments also evidenced that incorporating a data mining technique into a metaheuristic accelerated the process of finding near optimal and optimal solutions.

1 Introduction.

Metaheuristics represent an important class of approximate techniques for solving hard combinatorial optimization problems, for which the use of exact methods is impractical. They are general purpose high-level procedures that can be instantiated to explore efficiently the solution space of a specific optimization problem. Over the last decades, metaheuristics, like genetic algorithms, tabu search, simulated annealing, ant systems, GRASP, and others, have been proposed and applied to real-life problems of several areas of science [13]. An overview of heuristic search can be found in [17].

A trend in metaheuristics research is the exploration of hybrid metaheuristics [21]. One kind of such hybrid methods results from the combination of concepts and strategies behind two or more metaheuristics, and another kind corresponds to metaheuristics combined with concepts and processes from other areas responsible for performing specific tasks that can improve the original method. An instance of the latter case, and the subject of this work, is a hybrid version of the GRASP metaheuristic that incorporates a data mining process, called DM-GRASP (Data Mining GRASP) [20].

The GRASP (Greedy Randomized Adaptive Search Procedures) metaheuristic [2, 3], since it was proposed, has been successfully applied to solve many optimization problems [4]. The solution search process employed by GRASP is performed iteratively and each iteration consists of two phases: construction and local search. A feasible solution is built in the construction phase, and then its neighborhood is explored by the local search in order to find a better solution. The result is the best solution found over all iterations.

Data mining refers to the automatic extraction of knowledge from datasets [9, 23]. The extracted knowledge, expressed in terms of patterns or rules, represents important features of the dataset at hand. Hence, data mining provides a means to better understand concepts implicit in raw data, which is fundamental in a decision making process.

The hybridization of GRASP with a data mining process was first introduced and applied to the set packing problem [15, 16]. The basic idea was that patterns found in good quality solutions could be used to guide the search, leading to a more effective exploration of the solution space. The resulting method, the DM-GRASP metaheuristic, achieved promising results not only in terms of solution quality but also in terms of execution time required to obtain good quality solutions. Afterwards, the method was evaluated on two other applications, namely, the maximum diversity problem [18] and

*Supported by CAPES, CNPq and FAPERJ grants.
†Fluminense Federal University – plastino@ic.uff.br.
‡Fluminense Federal University – efonseca@ic.uff.br.
§Fluminense Federal University – rfuchshuber@ic.uff.br.
¶Fluminense Federal University – simone@ic.uff.br.
‖University of Kent – a.a.freitas@kent.ac.uk.
**University of Kent – ml86@kent.ac.uk.
††University of Kent – s.salhi@kent.ac.uk.

the server replication for reliable multicast problem [19], and the results were equally successful.

The first contribution of this work is the development and evaluation of a DM-GRASP implementation for the p-median problem. We intend to show that this optimization problem can also benefit from the idea of introducing a data mining module into a metaheuristic. The p-median is a well-known NP-hard problem [10], with important applications to real-life location problems [22], and can be generally stated as follows. Given a set F of m potential facilities and a set of n customers, the p-median problem consists of finding a subset of F with p facilities such that the cost of serving all customers is minimized. Many recent procedures developed to solve this problem are based on metaheuristics and GRASP procedures have achieved excellent results [11].

The DM-GRASP implementations successfully used for the set packing problem, the maximum diversity problem, and for the reliable multicast problem were developed over a common framework, divided in two parts. In the first one, a number of GRASP iterations are executed and the best solutions are stored in an elite set. Then, a data mining algorithm is used to extract patterns from this set of sub-optimal solutions. In the second part, the GRASP iterations use the mined patterns to construct new solutions. In this framework, the data mining process is performed just once, after exactly half of the GRASP iterations.

According to the taxonomy of hybrid metaheuristics proposed in [21], the DM-GRASP framework can be classified as a high-level and relay hybrid metaheuristic. It is considered high-level since the data mining technique and GRASP are self-contained and it is a relay hybridization because GRASP, the data mining process, and GRASP again are applied in a pipeline fashion.

Although good results have been achieved using this framework, another important contribution of this work is to evaluate how many times and at which moments the data mining process should be performed. We believe that mining more than once, and as soon as the elite set is stable and good enough, can improve the DM-GRASP framework. Based on this observation, in this work we also propose and evaluate another version of the DM-GRASP for the p-median problem, called MDM-GRASP (Multi Data Mining GRASP).

The remaining of this paper is organized as follows. In Section 2, we review the main concepts and the structure of the GRASP metaheuristic. In Section 3, we present the p-median problem and the GRASP implementation for this problem. The hybridization of GRASP, proposed in this work for the p-median problem, is defined in Section 4. The experiments conducted to compare the traditional GRASP and the Hybrid DM-GRASP are reported and discussed in Section 5. In Section 6, the multi data mining strategy is proposed and, in Section 7, this new approach is experimentally compared to the DM-GRASP. Finally, in Section 8, concluding remarks are made and some future works are pointed out.

2 The GRASP Metaheuristic.

GRASP [14] is a metaheuristic already applied successfully to many optimization problems [4]. It is a two-phase iterative process. The first phase of a GRASP iteration is the construction phase, in which a complete solution is built. Since this solution is not guaranteed to be locally optimal, a local search is performed in the second phase. This iterative process is repeated until a termination criterion is met and the best solution found over all iterations is taken as result.

A pseudo-code of the GRASP process is illustrated in Figure 1. In line 1, the variable that stores the best solution found is initialized. The block of instructions between lines 2 and 8 are executed iteratively. The construction phase is executed in line 3 and, in line 4, the local search is applied to the constructed solution. In line 5, the quality of the obtained solution is compared to the current best found and, if necessary, the best solution is updated. In line 9, the best solution is returned.

```
procedure GRASP()
1.  best_sol ← ∅;
2.  repeat
3.     sol ← Construction_Phase();
4.     sol ← Local_Search_Phase(sol);
5.     if Quality(sol) > Quality(best_sol)
6.        best_sol ← sol;
7.     end if
8.  until TerminationCriterion();
9.  return best_sol;
```

Figure 1: Pseudo-code of the GRASP metaheuristic

In the construction phase, the components of the solutions are selected one by one and incorporated into the partial solution until it is completely built. This process is illustrated in Figure 2. In line 1, the solution starts as an empty set. In each step executed from line 2 to line 6, the components not yet in the solution are ranked according to a greedy function. The better ranked components form a list, called Restricted Candidate List (RCL), in line 3. Suppose that, in a maximization problem, the best ranked component has the value max_value. The RCL can be, for example, composed by all components whose values are in the interval $[\alpha * max\_value, max\_value]$, where $\alpha \in [0, 1]$ is the parameter that, in general, defines the size of this list. In line 4, one component is randomly selected from this list and incorporated into the current solution in line 5. Note that this process would be purely greedy if the RCL was always composed only by the best component ($\alpha = 1$), and purely random if it was always composed by all possible components ($\alpha = 0$). In line 7, the complete solution is returned.

```
procedure Construction_Phase()
1.   sol ← ∅;
2.   repeat
3.     RCL ← BuildRCL(sol);
4.     s ← SelectRandom(RCL);
5.     sol ← sol ∪ {s};
6.   until SolutionCompleted(sol);
7.   return sol;
```

Figure 2: Pseudo-code of the construction phase

The solution obtained in the construction phase is not guaranteed to be locally optimal and becomes the starting point for the local search phase. Local search is a hill-climbing process, in which the neighborhood of the solution is explored. The neighborhood of a solution is defined by a function that relates this solution with a set of other solutions. If a better solution is found, the local search is performed again, considering the neighborhood of this new solution. Otherwise, the local search terminates.

Greedy methods usually find local optimal solutions due to the over intensive exploration of a small part of the solution space. In order to increase the probability of finding a global optimal solution, the solution search process must diversify the solution space regions to be explored. The GRASP metaheuristic can be understood as a method that is partially greedy and partially random, which leads to an effective exploration of the solution search space due to the satisfactory balance of search intensification and diversification.

3 GRASP for the p-Median Problem.

3.1 The p-Median Problem. The p-median is a well-known NP-hard problem [10] and has been employed in many important real-life location problems [22]. It can be stated as follows.

Let F be a set of m potential facilities and C a set of n customers. Let $d : C \times F \rightarrow \Re$ be a function which evaluates the distance between a customer and a potential facility. Given a positive integer p, $p \leq m$, the p-median problem consists of identifying a subset R of F such that $|R| = p$ and the sum of the distances from each customer in C to its closest facility in R is minimized.

Without loss of generality, in this work we will consider $F = C$, that is, in every customer location, there is a potential facility.

An instance of the p-median problem can then be represented by a constant p and a complete, weighted and undirected graph $G = (C, D)$, where $C = \{c_1, c_2, ..., c_n\}$ is the set of n customers, and the weight w_{ij} of each edge (c_i, c_j) in D represents the distance $d(c_i, c_j)$ between c_i and c_j.

A solution of an instance of the p-median problem, defined by p and $G = (C, D)$, can then be represented by a subset $R = \{r_1, r_2, ..., r_p\}$ of C.

Every solution R naturally partitions the set C into p clusters $P_1, P_2, ..., P_p$, such that each cluster P_i, $1 \leq i \leq p$, is composed by all customers whose closest facility is r_i.

3.2 The GRASP Implementation. As mentioned earlier, a GRASP implementation to solve a combinatorial optimization problem is generally composed by a construction and a local search procedures.

Figure 3 contains the pseudo-code of the GRASP construction phase implemented in this work. Initially, in line 1, the variable *sol*, which stores the best solution found is initialized. In line 2, all potential facilities in C are inserted into the candidate list CL, which stores all elements that can be part of the solution. From line 3 to 11, the construction iterations are executed until the solution is completed with p elements. In each iteration, an element is inserted into the solution. In lines 4 and 5, each element e in CL is evaluated by a greedy function that calculates the cost of the partial solution after the insertion of the element e. In line 7, the restricted candidate list RCL is generated with all elements in CL whose returned evaluations are in the interval $[mic, mic + (mac - mic) * \alpha]$, where mic and mac are the worst and the best returned values. In line 8, an element s is randomly selected from RCL and, in line 9, it is inserted into the solution sol. In line 10, the CL is updated. Finally, in line 12, the best solution found is returned.

In Figure 4, the pseudo-code of the GRASP local search is presented. Initially, in lines 1 to 2, some control variables are initialized. The function $cost\_eval()$, used in line 2, evaluates the cost of a solution by computing the sum of the distances between all customers and their closest facilities. From line 3 to 23, the neighborhood of the current solution is visited and if a better solution is found, it becomes the current one, which starts this process again, until no more improvement is made. The best solution found is returned in line 24. From line 5 to 22, p iterations are executed. In each iteration, from line 8 to 15, one element r_i of the solution is exchanged by all elements close to it in its partition (cluster) P_i. We consider that an element e is close to

```
procedure Construction_p_Median()
sol ← ∅;
CL ← C;
repeat
  for each element e in CL
    cost[e] ← sum of distances between each
              element in C − {sol ∪ {e}} and
              its closest element in sol ∪ {e};
  end for;
  RCL ← {e ∈ CL | cost[e] ∈ [mic, mic + (mac − mic) ∗ α]};
  s ← element randomly selected from RCL;
  sol ← sol ∪ {s};
  CL ← CL − {s};
until sol has p elements;
return sol;
```

Figure 3: Construction phase for the p-median problem

```
procedure Local_Search_p_Median(sol)
1.  best_sol ← sol;
2.  best_cost ← cost_eval(sol);
3.  repeat
4.    no_improvements ← true;
5.    for i = 1 to p
6.      approx_best_sol ← ∅;
7.      approx_best_cost ← ∞;
8.      for each element e in P_i close to r_i
9.        approx_sol ← exchange(best_sol, r_i, e);
10.       approx_cost ← approx_cost_eval(approx_sol);
11.       if approx_cost < approx_best_cost then
12.         approx_best_sol ← approx_sol;
13.         approx_best_cost ← approx_cost;
14.       end if
15.     end for;
16.     exact_sol_cost ← cost_eval(approx_best_sol);
17.     if exact_sol_cost < best_cost then
18.       best_sol ← approx_best_sol;
19.       best_cost ← exact_sol_cost;
20.       no_improvements ← false;
21.     end if
22.   end for;
23. until no_improvements;
24. return best_sol;
```

Figure 4: Local search phase for the p-median problem

r_i in its partition P_i if the distance between e and r_i is less or equal to the average of distances between r_i and all elements in P_i. In order to reduce the computational effort of the local search, the solution obtained by each exchange is approximately evaluated in line 10 and only the best one is exactly evaluated in line 16. The function *approx_cost_eval*(), used in line 10, evaluates approximately the cost of a solution by recalculating the distances only within the partition P_i, without making this computation within the other partitions, which would be necessary for the exact calculation, since there was a change of location. In lines from 17 to 21, it is verified if a better solution than the current one was reached. If so, this new one becomes the current solution and the local search starts again.

4 DM-GRASP for the p-Median Problem.

In this section, we present the main contribution of this work: The hybrid version of the GRASP metaheuristic which incorporates a data mining process, called DM-GRASP, to solve the p-median problem.

In the original GRASP, iterations are performed independently and, consequently, the knowledge acquired in past iterations is not exploited in subsequent iterations. The basic concept of incorporating a data mining process in GRASP is that patterns found in high quality solutions obtained in earlier iterations can be used to conduct and improve the search process.

The DM-GRASP is composed of two phases. The first one is called the elite set generation phase, which consists of executing n pure GRASP iterations to obtain a set of different solutions. The d best solutions from this set of solutions compose the elite set.

After this first phase, the data mining process is applied. It is responsible for extracting patterns from the elite set. The patterns to be mined are sets of elements that frequently appear in solutions from the elite set. This extraction of patterns characterizes a frequent itemset mining application [9]. A frequent itemset mined with support $s\%$ represents a set of elements that occur in $s\%$ of the elite solutions.

Next, the second phase, called hybrid phase, is performed. In this part, another n slightly different GRASP iterations are executed. In these n iterations, an adapted construction phase starts building a solution guided by a mined pattern selected from the set of mined patterns. Initially, all elements of the selected pattern are inserted into the partial solution, from which a complete solution will be built executing the standard construction procedure. This way, all constructed solutions will contain the elements of the selected pattern.

The pseudo-code of the DM-GRASP for the p-median problem is illustrated in Figure 5. In lines 1 and 2, the best solution and the elite set are initialized with the empty set. The loop from line 3 to line 10 corresponds to the elite set generation phase, in which pure GRASP is performed for n iterations. The original construction method is executed in line 4, followed by the local search method in line 5. The elite set, composed of d solutions, is updated in line 6. In line 7, it is checked whether the best solution should be updated, which is done in line 8. The data mining procedure extracts the set of patterns from the elite set in line 11. The loop from line 12 to line 19 corresponds to the hybrid phase. Each iteration is based on a pattern selected in line 13. The largest pattern is chosen. If there are more than one largest pattern, one of them is randomly selected. The adapted construction procedure is performed in line 14, using the selected pattern as a starting point. In line 15, the local search is executed. From line 16 to 18 the best solution is updated. The best solution is returned in line 20.

```
procedure DM_GRASP_p_Median()
 1.  best_sol ← ∅;
 2.  elite_set ← ∅;
 3.  for it ← 1 to n do
 4.    sol ← Construction_p_Median();
 5.    sol ← Local_Search_p_Median(sol);
 6.    UpdateElite(elite_set, sol, d);
 7.    if Cost(sol) < Cost(best_sol)
 8.      best_sol ← sol;
 9.    end if
10.  end for
11.  patterns_set ← Mine(elite_set);
12.  for it ← 1 to n do
13.    pattern ← SelectNextLargestPattern(patterns_set);
14.    sol ← Adapted_Construction_p_Median(pattern);
15.    sol ← LocalSearch_Phase(sol);
16.    if Cost(sol) < Cost(best_sol)
17.      best_sol ← sol;
18.    end if
19.  end for;
20.  return best_sol;
```

Figure 5: Pseudo-code of the DM-GRASP

In Figure 6, the pseudo-code of the adapted construction is illustrated. It is quite similar to the code described in Figure 3 with the difference that, instead of beginning the solution with an empty set, in line 1, it starts with all elements of the pattern supplied as a parameter. In line 2, these elements already inserted in the solution are removed from the candidate list CL.

```
procedure Adapted_Construction_p_Median(pattern)
 1. sol ← pattern;
 2. CL ← C − pattern;
 3. repeat
 4.   for each element e in CL
 5.     cost[e] ← sum of distances between each
                  element in C − {sol ∪ {e}} and
                  its closest element in sol ∪ {e};
 6.   end for;
 7.   RCL ← {e ∈ CL|cost[e] ∈ [mic, mic + (mac − mic) ∗ α]};
 8.   s ← element randomly selected from RCL;
 9.   sol ← sol ∪ {s};
10.   CL ← CL − {s};
11. until sol has p elements;
12. return sol;
```

Figure 6: Pseudo-code of the adapted construction

The extraction of patterns from the elite set, which is activated in line 11 of the pseudo-code presented in Figure 5, corresponds to the well-known frequent itemset mining (FIM) task. The FIM problem can be defined as follows.

Let $I = \{i_1, i_2, ..., i_n\}$ be a set of items. A transaction t is a subset of I and a dataset D is a set of transactions. A frequent itemset F, with support s, is a subset of I which occurs in at least $s\%$ of the transactions in D. The FIM problem consists of extracting all frequent itemset from a dataset D with a minimum support specified as a parameter. During the last fifteen years, many algorithms have been proposed to efficiently mine frequent itemsets [1, 6, 8, 12].

In this work, the useful patterns to be mined are sets of elements that commonly appear in sub-optimal solutions of the p-median problem. This is a typical frequent itemset mining application, where the set of items is the set of potential locations. Each transaction of the dataset represents a sub-optimal solution of the elite set. A frequent itemset mined from the elite set with support $s\%$ represents a set of locations that occur in $s\%$ of the elite solutions.

A frequent itemset is called maximal if it has no superset that is also frequent. In order to avoid mining frequent itemsets which are subset of one another, in the DM-GRASP proposal for the p-median problem, we decided to extract only maximal frequent itemset. To execute this task, we adopted the FPmax* algorithm [7], available at http://fimi.cs.helsinki.fi.

5 Computational Results for DM-GRASP.

In this section, the computational results obtained for GRASP and DM-GRASP are presented. The 80 p-median problem instances adopted in the conducted experiments were also used in [5]. There are four groups with 20 instances each: instances in the first group ($G50$) have 50 customers and p varying from 6 to 25 with an increment of 1. In the second ($G287$), third ($G654$), and fourth ($G1060$) groups, instances have 287, 654, and 1060 customers, respectively, with p varying from 5 to 100 with an increment of 5. For all instances the optimal values are known.

The algorithms were implemented in C++ and compiled with g++ (GCC) 4.2.3. The tests were performed on a 2.4 GHz Intel Core 2 Quad CPU Q6600 with 3 Gbytes of RAM, running Linux Kernel 2.6.24.

Both GRASP and DM-GRASP were run 10 times with a different random seed in each run. The parameter α was set to 0.2 and each strategy executed $max(500, min(3 * nc, 2000))$ iterations, where nc is the number of customers of the instance. Different numbers of iterations allow more search for larger instances.

In Tables 1, 2, 3, and 4, the results related to the quality of the obtained solutions are shown. The first column presents the value of p, the second one shows the optimal value for this instance – obtained by ILOG CPLEX –, the third and fifth columns present the deviation value of the best cost obtained by GRASP and DM-GRASP related to the optimal value, and the fourth and sixth columns present the deviation value of the average cost obtained by GRASP and DM-GRASP.

The deviation value is computed as follows:

$$dev = \frac{(HeuristicCost - OptCost)}{OptCost} \times 100, \tag{5.1}$$

where $HeuristicCost$ is the (best or average) cost obtained by the heuristic technique and the $OptCost$ is the optimal value for the working instance.

In each table, the smallest values, i.e., the better results, are bold-faced. The last line of these four tables presents the average values of each column. In all tables, the average results obtained by DM-GRASP are better than the results obtained by GRASP, except for the average value of the best results in Table 1.

Table 1 shows that 18 optimal solutions were found by both DM-GRASP and GRASP. DM-GRASP found 10 better results for average deviation and GRASP found 4. Table 2 shows that DM-GRASP found 18 better results for best deviation and 2 were found by GRASP. DM-GRASP found better results for average deviation in all instances. Table 3 shows that DM-GRASP found 13 better results for best deviation and 2 were found by GRASP. DM-GRASP found 16 better results for average deviation and GRASP just one. Table 4 shows that DM-GRASP found 17 better results for best deviation and 18 better results for average deviation values, and GRASP found no better values.

These results show that the proposed DM-GRASP strategy was able to improve the results obtained by GRASP in the large majority of the cases.

Tables 5, 6, 7, and 8 present the results related to execution time of both strategies. In these tables, the first column presents the value of p, the second and fourth columns show the average execution time (in seconds) of GRASP and DM-GRASP, obtained for 10 runs, the third and fifth columns present the standard deviation value of these execution times. The sixth column shows the percentual difference between the GRASP and DM-GRASP average times in relation to the GRASP average time.

For all Tables, the execution times for DM-GRASP are considerably smaller than those for GRASP. The standard deviations are quite small, which shows the robustness of DM-GRASP.

There are two main reasons for the faster behavior of DM-GRASP. First, the computational effort of the adapted construction phase is smaller than the tradi-

		GRASP		DM-GRASP	
p	Optimal	Best	Avg	Best	Avg
6	61.31	0.00	0.00	0.00	0.00
7	55.15	0.00	0.00	0.00	0.00
8	50.26	0.00	0.00	0.00	0.00
9	46.26	0.00	0.00	0.00	0.00
10	42.37	0.00	0.00	0.00	0.00
11	38.80	0.00	0.00	0.00	0.00
12	35.53	0.00	0.08	0.00	**0.00**
13	32.77	0.00	**0.09**	0.00	0.12
14	30.12	0.00	0.66	0.00	**0.00**
15	27.99	0.00	0.36	0.00	**0.04**
16	26.00	0.00	0.65	0.00	**0.35**
17	24.21	0.00	**0.33**	0.00	0.37
18	22.46	0.00	0.71	0.00	**0.40**
19	20.73	0.00	1.83	0.00	**0.87**
20	19.47	0.00	0.87	0.00	**0.72**
21	18.21	0.22	1.15	0.22	**0.60**
22	17.00	0.00	**0.71**	0.00	0.76
23	15.83	**0.00**	**0.51**	0.38	0.95
24	14.65	0.14	1.71	**0.00**	**1.30**
25	13.51	0.00	2.00	0.00	**1.26**
Average		**0.02**	0.58	0.03	**0.39**

Table 1: GRASP and DM-GRASP for group G50

		GRASP		DM-GRASP	
p	Optimal	Best	Avg	Best	Avg
5	9715.63	3.35	6.86	**0.86**	**4.35**
10	6757.13	6.36	10.01	**3.35**	**6.92**
15	5237.04	9.76	12.24	**6.47**	**8.86**
20	4167.50	9.71	14.22	**3.94**	**9.44**
25	3359.60	**5.15**	10.66	6.38	**9.98**
30	2723.51	**2.43**	6.82	3.99	**6.09**
35	2246.57	4.93	7.44	**2.15**	**5.73**
40	1909.70	7.02	8.50	**4.30**	**6.49**
45	1636.22	4.81	6.40	**3.64**	**5.47**
50	1405.30	4.84	6.83	**3.04**	**4.85**
55	1206.85	5.30	7.76	**4.36**	**6.27**
60	1057.71	6.81	8.18	**3.20**	**4.95**
65	928.97	5.19	7.42	**3.05**	**5.21**
70	817.41	6.33	8.22	**3.09**	**5.41**
75	731.12	6.15	8.81	**4.47**	**6.41**
80	656.75	5.78	8.69	**3.09**	**5.51**
85	589.33	7.66	11.26	**4.57**	**6.50**
90	529.95	9.73	11.64	**2.99**	**5.67**
95	481.53	8.94	11.08	**2.65**	**5.29**
100	441.95	8.70	10.10	**1.78**	**4.10**
Average		6.45	9.16	**3.57**	**6.17**

Table 2: GRASP and DM-GRASP for group G287

		GRASP		DM-GRASP	
p	Optimal	Best	Avg	Best	Avg
5	209155.00	0.00	0.00	0.00	0.00
10	115789.00	0.00	0.00	0.00	0.00
15	80595.40	0.00	0.00	0.00	0.00
20	63894.70	0.00	**0.01**	0.00	0.02
25	52875.80	0.24	0.61	**0.02**	**0.15**
30	45307.10	0.10	0.43	**0.01**	**0.09**
35	39862.00	1.17	2.11	**0.31**	**0.80**
40	36228.30	0.57	1.44	**0.33**	**0.76**
45	32779.10	0.84	2.00	0.84	**1.82**
50	29774.10	2.42	2.80	**2.07**	**2.44**
55	27200.10	**2.40**	3.62	2.52	**3.11**
60	24984.00	3.71	4.49	**2.48**	**3.26**
65	23129.20	3.77	4.73	**2.13**	**3.10**
70	21851.80	3.04	3.88	**2.26**	**2.77**
75	20740.40	1.82	3.33	**0.99**	**1.77**
80	19748.20	2.11	2.83	**1.28**	**1.70**
85	18837.20	2.17	2.65	**1.57**	**1.91**
90	18008.40	**1.66**	2.66	1.73	**2.00**
95	17259.80	2.43	2.80	**1.51**	**2.03**
100	16544.10	1.97	2.91	**1.52**	**2.03**
Average		1.52	2.16	**1.08**	**1.49**

Table 3: GRASP and DM-GRASP for group G654

		GRASP		DM-GRASP	
p	Optimal	Best	Avg	Best	Avg
5	1854330.00	0.00	0.00	0.00	0.00
10	1252140.00	0.00	0.00	0.00	0.00
15	982399.00	0.00	0.01	0.00	**0.00**
20	831419.00	0.06	0.19	**0.00**	**0.02**
25	725006.00	0.15	0.34	**0.00**	**0.07**
30	641851.00	0.16	0.51	**0.07**	**0.16**
35	581365.00	0.31	0.79	**0.19**	**0.31**
40	532432.00	0.65	1.11	**0.43**	**0.63**
45	492476.00	0.72	1.08	**0.35**	**0.58**
50	455588.00	1.35	1.56	**0.32**	**0.83**
55	425178.00	0.97	1.66	**0.16**	**0.58**
60	399964.00	1.07	1.67	**0.34**	**0.81**
65	379072.00	1.11	1.55	**0.26**	**0.69**
70	360127.00	1.51	1.91	**0.21**	**0.78**
75	343261.00	1.67	1.97	**0.87**	**1.17**
80	329073.00	1.92	2.22	**0.91**	**1.16**
85	316450.00	1.79	2.13	**0.77**	**1.18**
90	304974.00	2.11	2.46	**1.19**	**1.57**
95	294660.00	2.25	2.57	**1.20**	**1.65**
100	284815.00	2.41	2.69	**1.72**	**1.91**
Average		1.01	1.32	**0.45**	**0.71**

Table 4: GRASP and DM-GRASP for group G1060

	GRASP		DM-GRASP		
p	Time (s)	SD	Time (s)	SD	%
6	0.09	0.01	**0.07**	0.01	22.22
7	0.13	0.01	**0.08**	0.02	38.46
8	0.14	0.02	**0.09**	0.01	35.71
9	0.18	0.01	**0.11**	0.01	38.89
10	0.20	0.03	**0.11**	0.01	45.00
11	0.21	0.03	**0.12**	0.01	42.86
12	0.22	0.01	**0.12**	0.02	45.45
13	0.24	0.01	**0.13**	0.01	45.83
14	0.23	0.01	**0.13**	0.01	43.48
15	0.26	0.03	**0.14**	0.01	46.15
16	0.27	0.02	**0.14**	0.01	48.15
17	0.28	0.02	**0.14**	0.01	50.00
18	0.27	0.01	**0.15**	0.01	44.44
19	0.28	0.03	**0.16**	0.01	42.86
20	0.29	0.02	**0.16**	0.02	44.83
21	0.29	0.04	**0.15**	0.01	48.28
22	0.32	0.04	**0.15**	0.01	53.12
23	0.31	0.04	**0.16**	0.01	48.39
24	0.33	0.10	**0.16**	0.02	51.52
25	0.31	0.04	**0.15**	0.01	51.61

Table 5: Time of GRASP and DM-GRASP for G50

tional construction, since the elements from a pattern are initially fixed in the solution. Then a smaller number of elements must be processed and inserted into the constructed solution. Second, the use of patterns leads to the construction of better solutions which will be input for the local search. This incurs in less computational effort taken to converge to a local optimal solution.

6 MDM-GRASP for the p-Median Problem.

The computational experiments reported in the previous section showed that the introduction of the data mining process allowed GRASP to find better solutions in less computational time. In the proposed hybrid GRASP, the data mining procedure is executed just once and at the middle point of the whole process. Although the obtained results were satisfactory, we believe that mining more than once, and as soon as the elite set is stable and good enough, can improve the original DM-GRASP framework. Based on this hypothesis, in this work we also propose and evaluate another version of the DM-GRASP for the p-median problem, called MDM-GRASP (Multi Data Mining GRASP).

The main idea of this proposal is to execute the mining process: (a) as soon as the elite set becomes stable – which means that no change in the elite set occurs throughout a given number of iterations – and (b) whenever the elite set has been changed and again has become stable. We hypothesize that mining more

	GRASP		DM-GRASP		
p	Time (s)	SD	Time (s)	SD	%
5	1.76	0.02	**1.58**	0.08	10.23
10	3.65	0.05	**3.12**	0.08	14.52
15	5.57	0.04	**4.56**	0.10	18.13
20	7.47	0.08	**5.97**	0.15	20.08
25	9.21	0.06	**7.21**	0.20	21.72
30	10.95	0.08	**8.04**	0.14	26.58
35	12.62	0.05	**9.28**	0.18	26.47
40	14.32	0.05	**10.59**	0.16	26.05
45	16.03	0.09	**11.69**	0.15	27.07
50	17.68	0.06	**12.97**	0.17	26.64
55	19.47	0.09	**14.36**	0.12	26.25
60	21.18	0.05	**15.77**	0.13	25.54
65	22.87	0.04	**17.11**	0.16	25.19
70	24.45	0.06	**18.33**	0.09	25.03
75	26.19	0.23	**19.54**	0.23	25.39
80	27.50	0.07	**20.41**	0.22	25.78
85	28.78	0.07	**21.11**	0.27	26.65
90	29.91	0.09	**22.10**	0.36	26.11
95	30.99	0.10	**22.66**	0.31	26.88
100	31.92	0.11	**23.31**	0.19	26.97

Table 6: Time of GRASP and DM-GRASP for G287

	GRASP		DM-GRASP		
p	Time (s)	SD	Time (s)	SD	%
5	14.54	0.15	**12.53**	0.30	13.82
10	31.02	0.22	**19.92**	0.08	35.78
15	47.84	0.17	**31.27**	0.85	34.64
20	65.01	0.22	**41.00**	0.86	36.93
25	83.06	0.39	**57.12**	1.02	31.23
30	101.59	0.31	**74.20**	1.40	26.96
35	120.87	0.54	**95.22**	1.79	21.22
40	139.70	0.46	**111.90**	1.46	19.90
45	160.03	0.73	**128.16**	1.29	19.92
50	180.64	0.32	**141.89**	1.70	21.45
55	202.27	0.27	**158.68**	1.33	21.55
60	223.24	0.43	**173.04**	2.02	22.49
65	243.09	0.47	**188.85**	0.97	22.31
70	262.74	0.95	**204.60**	1.56	22.13
75	280.63	0.48	**221.63**	2.61	21.02
80	298.97	0.70	**238.20**	1.43	20.33
85	317.94	0.53	**254.31**	2.11	20.01
90	337.88	0.55	**271.90**	1.49	19.53
95	358.21	0.64	**289.15**	3.64	19.28
100	377.81	0.64	**303.49**	2.56	19.67

Table 7: Time of GRASP and DM-GRASP for G654

	GRASP		DM-GRASP		
p	Time (s)	SD	Time (s)	SD	%
5	70.82	2.38	**33.28**	1.96	53.01
10	151.69	4.73	**62.03**	0.55	59.11
15	227.93	3.57	**89.05**	4.18	60.93
20	309.00	4.89	**138.92**	4.05	55.04
25	382.88	7.41	**182.04**	4.70	52.46
30	457.96	6.26	**220.87**	4.78	51.77
35	535.30	4.45	**261.29**	6.11	51.19
40	618.57	4.19	**311.84**	4.42	49.59
45	701.37	5.80	**357.01**	4.95	49.10
50	786.00	4.19	**398.47**	8.21	49.30
55	882.14	6.41	**440.81**	7.06	50.03
60	971.43	6.76	**486.93**	6.40	49.87
65	1052.06	8.08	**538.06**	6.93	48.86
70	1144.24	6.66	**585.01**	6.29	48.87
75	1233.64	7.97	**633.64**	6.83	48.64
80	1334.10	6.47	**681.62**	8.21	48.91
85	1422.74	4.80	**736.56**	8.51	48.23
90	1506.16	7.56	**795.74**	9.46	47.17
95	1573.72	18.11	**852.16**	14.77	45.85
100	1742.93	37.33	**900.21**	10.45	48.35

Table 8: Time of GRASP and DM-GRASP for G1060

than once will explore the gradual evolution of the elite set and allow the extraction of refined patterns.

The pseudo-code of the MDM-GRASP for the p-median problem is illustrated in Figure 7. In lines 1 and 2, the best solution and the elite set are initialized with the empty set. The loop from line 3 to 10 corresponds to the first elite set generation phase, in which pure GRASP iterations are performed until the elite set becomes ready to be mined or the termination criterion – the total number of iterations – becomes true. In the current implementation, we consider that the elite set is ready if it has not being changed for 5% of the total number of iterations after has been updated. Next, in the loop from line 11 to 22, whenever the elite set is ready, the data mining procedure is executed and extracts a new pattern set in line 13 and, from line 15 to 21, a hybrid iteration is executed. In line 15, the next largest pattern is selected. If there are more than one largest pattern, they are randomly selected. Then the adapted construction is performed in line 16, using the selected pattern as a starting point. In line 17, the local search is executed. From line 19 to 21, the best solution is updated. The best solution is returned in line 23.

7 Computational Results for MDM-GRASP.

In this section, the computational results obtained for the proposed MDM-GRASP strategy are compared to the previously reported DM-GRASP results. The p-median problem instances are the same used in the

```
procedure MDM_GRASP_p_Median()
    best_sol ← ∅;
    elite_set ← ∅;
    repeat
      sol ← Construction_p_Median();
      sol ← Local_Search_p_Median(sol);
      UpdateElite(elite_set, sol, d);
      if Cost(sol) < Cost(best_sol)
        best_sol ← sol;
      end if
    until elite_set_is_ready or end_criterion;
    while not end_criterion;
      if elite_set_is_ready
        patterns_set ← Mine(elite_set);
      end if
      pattern ← SelectNextLargestPattern(patterns_set);
      sol ← Adapted_Construction_p_Median(pattern);
      sol ← LocalSearch_Phase(sol);
      UpdateElite(elite_set, sol, d);
      if Cost(sol) < Cost(best_sol)
        best_sol ← sol;
      end if
    end while;
    return best_sol;
```

Figure 7: Pseudo-code of the MDM-GRASP

previous section. The MDM-GRASP was also run 10 times with a different random seed in each run. The parameter α and the number of executed iterations were also the same used in the previous experiments.

In Tables 9, 10, 11, and 12, the results related to quality of the obtained solutions are shown. In all Tables, the average results obtained by MDM-GRASP are better than the results obtained by DM-GRASP.

Table 9 shows that 19 optimal solutions were found by MDM-GRASP and 18 were found by DM-GRASP. MDM-GRASP found 12 better results for average deviation and DM-GRASP did not find any. Table 10 shows that MDM-GRASP found 16 better results for best deviation and 4 were found by DM-GRASP. MDM-GRASP found 19 better results for average deviation and DM-GRASP just one. Table 11 shows that MDM-GRASP found 13 better results for best deviation and DM-GRASP just one. MDM-GRASP found 17 better results for average deviation and DM-GRASP did not find any. Table 12 shows that MDM-GRASP found 14 better results for best deviation and 1 was found by DM-GRASP. MDM-GRASP found 17 better results for average deviation, and DM-GRASP found no better result. These results show that the MDM-GRASP proposal was able to improve the results obtained by DM-GRASP.

Tables 13 and 14 compare the execution times spent by DM-GRASP and MDM-GRASP for the G654 and G1060 instances. The results for instances from groups G50 and G287 are similar. We can note that the

		DM-GRASP		MDM-GRASP	
p	Optimal	Best	Avg	Best	Avg
6	61.31	0.00	0.00	0.00	0.00
7	55.15	0.00	0.00	0.00	0.00
8	50.26	0.00	0.00	0.00	0.00
9	46.26	0.00	0.00	0.00	0.00
10	42.37	0.00	0.00	0.00	0.00
11	38.80	0.00	0.00	0.00	0.00
12	35.53	0.00	0.00	0.00	0.00
13	32.77	0.00	0.12	0.00	**0.00**
14	30.12	0.00	0.00	0.00	0.00
15	27.99	0.00	0.04	0.00	**0.00**
16	26.00	0.00	0.35	0.00	**0.08**
17	24.21	0.00	0.37	0.00	**0.04**
18	22.46	0.00	0.40	0.00	**0.18**
19	20.73	0.00	0.87	0.00	**0.63**
20	19.47	0.00	0.72	0.00	**0.36**
21	18.21	0.22	0.60	**0.00**	**0.49**
22	17.00	0.00	0.76	0.00	**0.59**
23	15.83	0.38	0.95	**0.13**	**0.69**
24	14.65	0.00	1.30	0.00	**0.41**
25	13.51	0.00	1.26	0.00	**1.11**
Average		0.03	0.39	**0.01**	**0.23**

Table 9: DM and MDM-GRASP for group G50

		DM-GRASP		MDM-GRASP	
p	Optimal	Best	Avg	Best	Avg
5	9715.63	0.86	4.35	**0.38**	**3.68**
10	6757.13	3.35	6.92	**1.42**	**4.27**
15	5237.04	6.47	8.86	**5.45**	**7.24**
20	4167.50	**3.94**	9.44	5.16	**7.41**
25	3359.60	6.38	9.98	**2.71**	**7.91**
30	2723.51	3.99	**6.09**	**3.17**	6.27
35	2246.57	2.15	5.73	**2.09**	**4.80**
40	1909.70	4.30	6.49	**2.75**	**4.70**
45	1636.22	3.64	5.47	**2.82**	**4.34**
50	1405.30	3.04	4.85	**1.44**	**3.46**
55	1206.85	**4.36**	6.27	4.37	**5.54**
60	1057.71	3.20	4.95	**2.83**	**3.93**
65	928.97	**3.05**	5.21	3.55	**4.33**
70	817.41	3.09	5.41	**2.31**	**4.11**
75	731.12	4.47	6.41	**2.35**	**4.32**
80	656.75	**3.09**	5.51	3.54	**4.43**
85	589.33	4.57	6.50	**3.05**	**4.47**
90	529.95	2.99	5.67	**1.97**	**3.62**
95	481.53	2.65	5.29	**1.21**	**3.56**
100	441.95	1.78	4.10	**0.74**	**2.16**
Average		3.57	6.17	**2.67**	**4.73**

Table 10: DM and MDM-GRASP for group G287

		DM-GRASP		MDM-GRASP	
p	Optimal	Best	Avg	Best	Avg
5	209155.00	0.00	0.00	0.00	0.00
10	115789.00	0.00	0.00	0.00	0.00
15	80595.40	0.00	0.00	0.00	0.00
20	63894.70	0.00	0.02	0.00	**0.01**
25	52875.80	0.02	0.15	**0.00**	**0.10**
30	45307.10	0.01	0.09	0.01	**0.02**
35	39862.00	0.31	0.80	**0.29**	**0.50**
40	36228.30	0.33	0.76	**0.21**	**0.44**
45	32779.10	0.84	1.82	0.84	**1.22**
50	29774.10	2.07	2.44	**1.46**	**1.97**
55	27200.10	2.52	3.11	**1.36**	**2.08**
60	24984.00	2.48	3.26	**1.35**	**2.40**
65	23129.20	2.13	3.10	**0.97**	**2.18**
70	21851.80	2.26	2.77	**1.19**	**1.67**
75	20740.40	**0.99**	1.77	1.11	**1.48**
80	19748.20	1.28	1.70	**1.25**	**1.61**
85	18837.20	1.57	1.91	**1.07**	**1.48**
90	18008.40	1.73	2.00	**1.18**	**1.71**
95	17259.80	1.51	2.03	**1.02**	**1.61**
100	16544.10	1.52	2.03	**0.51**	**1.30**
Average		1.08	1.49	**0.69**	**1.09**

Table 11: DM and MDM-GRASP for group G654

		DM-GRASP		MDM-GRASP	
p	Optimal	Best	Avg	Best	Avg
5	1854330.00	0.00	0.00	0.00	0.00
10	1252140.00	0.00	0.00	0.00	0.00
15	982399.00	0.00	0.00	0.00	0.00
20	831419.00	0.00	0.02	0.00	**0.01**
25	725006.00	0.00	0.07	0.00	**0.01**
30	641851.00	0.07	0.16	**0.00**	**0.04**
35	581365.00	0.19	0.31	**0.04**	**0.17**
40	532432.00	0.43	0.63	**0.28**	**0.38**
45	492476.00	0.35	0.58	**0.21**	**0.40**
50	455588.00	0.32	0.83	**0.21**	**0.48**
55	425178.00	0.16	0.58	**0.15**	**0.37**
60	399964.00	0.34	0.81	**0.26**	**0.49**
65	379072.00	0.26	0.69	**0.19**	**0.51**
70	360127.00	**0.21**	0.78	0.22	**0.64**
75	343261.00	0.87	1.17	**0.55**	**0.78**
80	329073.00	0.91	1.16	**0.51**	**0.78**
85	316450.00	0.77	1.18	**0.65**	**0.89**
90	304974.00	1.19	1.57	**0.66**	**0.97**
95	294660.00	1.20	1.65	**1.11**	**1.27**
100	284815.00	1.72	1.91	**1.14**	**1.49**
Average		0.45	0.71	**0.31**	**0.48**

Table 12: DM and MDM-GRASP for group G1060

	DM-GRASP		MDM-GRASP		
p	Time (s)	SD	Time (s)	SD	%
5	12.53	0.30	**11.00**	0.22	12.21
10	19.92	0.08	**11.37**	0.45	42.92
15	31.27	0.85	**22.69**	2.50	27.44
20	41.00	0.86	**28.46**	3.02	30.59
25	57.12	1.02	**44.30**	4.36	22.44
30	74.20	1.40	**61.41**	4.71	17.24
35	95.22	1.79	**81.83**	3.77	14.06
40	111.90	1.46	**97.22**	10.62	13.12
45	128.16	1.29	**109.88**	3.81	14.26
50	141.89	1.70	**124.33**	7.75	12.38
55	158.68	1.33	**136.31**	7.32	14.10
60	173.04	2.02	**147.41**	9.26	14.81
65	188.85	0.97	**162.69**	13.43	13.85
70	204.60	1.56	**177.61**	15.45	13.19
75	221.63	2.61	**195.72**	6.81	11.69
80	238.20	1.43	**210.66**	12.56	11.56
85	254.31	2.11	**220.66**	15.27	13.23
90	271.90	1.49	**247.92**	25.19	8.82
95	289.15	3.64	**252.02**	18.79	12.84
100	303.49	2.56	**261.17**	18.10	13.94

Table 13: Time of DM and MDM-GRASP for G654

MDM-GRASP was faster than the DM-GRASP for all instances. The standard deviations are quite small, which also shows the robustness of MDM-GRASP.

The main reason for the faster behavior of MDM-GRASP is that the first execution of the data mining process occurs earlier than the unique data mining execution in the DM-GRASP strategy. Because of this, the benefit from using patterns in terms of execution time, which was already identified in the comparison between GRASP and DM-GRASP, was anticipated in the MDM-GRASP executions.

Figures 8, 9, and 10 present the behavior of the construction and local search phases, in terms of the cost values obtained, by GRASP, DM-GRASP, and MDM-GRASP throughout the execution of 2000 iterations, for a specific instance (p=80 in G1060).

Since the p-median is a minimization problem, the figures show that the local search always reduces the cost of the solution obtained by the construction phase. In Figure 8, we observe that GRASP maintains the behavior of both construction and local search during all the execution. In Figure 9, since the data mining procedure is executed after iteration 1000, the DM-GRASP strategy, from this point, makes an improvement in the quality of the solutions reached by the construction and local search phases. The behavior of MDM-GRASP is presented in Figure 10. There were four executions of the data mining procedures, after the iterations 584, 1023, 1378, and 1863. We can see then that the improvement due to the execution of the data mining pro-

	DM-GRASP		MDM-GRASP		
p	Time (s)	SD	Time (s)	SD	%
5	33.28	1.96	**25.84**	1.82	22.36
10	62.03	0.55	**48.26**	2.87	22.20
15	89.05	4.18	**67.58**	10.21	24.11
20	138.92	4.05	**119.16**	14.69	14.22
25	182.04	4.70	**155.57**	16.57	14.54
30	220.87	4.78	**180.24**	16.41	18.40
35	261.29	6.11	**220.43**	17.76	15.64
40	311.84	4.42	**263.32**	25.85	15.56
45	357.01	4.95	**315.17**	27.86	11.72
50	398.47	8.21	**329.08**	14.32	17.41
55	440.81	7.06	**377.18**	30.49	14.43
60	486.93	6.40	**405.63**	21.27	16.70
65	538.06	6.93	**461.86**	9.75	14.16
70	585.01	6.29	**520.64**	27.75	11.00
75	633.64	6.83	**537.92**	26.76	15.11
80	681.62	8.21	**574.46**	31.34	15.72
85	736.56	8.51	**616.94**	27.22	16.24
90	795.74	9.46	**656.35**	26.09	17.52
95	852.16	14.77	**743.59**	57.80	12.74
100	900.21	10.45	**816.66**	56.07	9.28

Table 14: Time of DM and MDM-GRASP for G1060

cess started to happen earlier. And, differently from the DM-GRASP, since patterns are extracted more than once, we can observe that the MDM-GRASP reduces gradually the cost of the solutions obtained by the construction and local search phases, which justify the good general behavior of this approach.

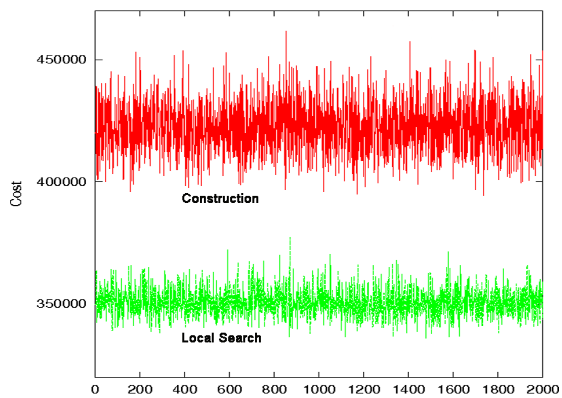

Figure 8: One execution of GRASP.

8 Statistical Significance of the Results.

In order to assess whether or not the differences of mean values obtained by the evaluated strategies are statistically significant, we employed the unpaired Student's t-test technique. Table 15 presents, for each pair of metaheuristics and for each instance group, the number of better average solutions found by each strategy

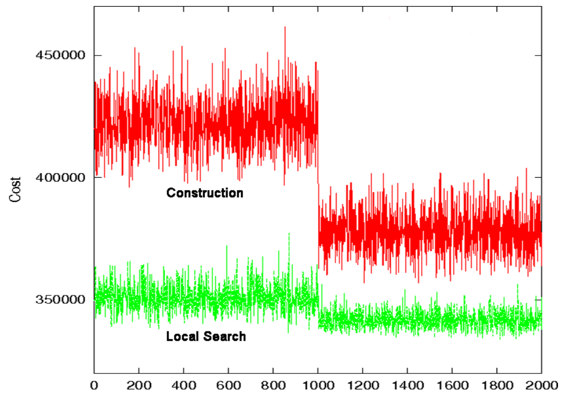

Figure 9: One execution of DM-GRASP.

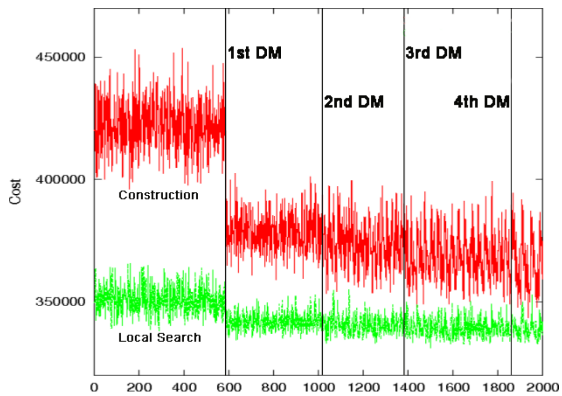

Figure 10: One execution of MDM-GRASP.

Strategy	Instance			
	G50	G287	G654	G1060
GRASP	4 (0)	0 (0)	1 (0)	0 (0)
DM-GRASP	10 (0)	20 (13)	16 (13)	18 (17)
DM-GRASP	0 (0)	1 (0)	0 (0)	0 (0)
MDM-GRASP	12 (0)	19 (6)	17 (9)	17 (11)
GRASP	1 (0)	0 (0)	0 (0)	0 (0)
MDM-GRASP	13 (2)	20 (17)	16 (16)	18 (17)

Table 15: t-test results with p-value equal to 0.01

and, between parentheses, the number among them that presents a p-value less than 0.01, which means that the probability of the difference of performance being due to random chance alone is less than 0.01.

For the group of 50 customers, almost all differences of performance are not significant as these instances seems to be easy to solve. For the other three groups, we can note that almost all results obtained by both DM-GRASP and MDM-GRASP, when compared to GRASP, are statistically significant. It should also be noted that the results of MDM-GRASP are in general considerably better than the results of DM-GRASP.

9 Conclusions.

In this work, we proposed a hybrid version of the GRASP metaheuristic, called DM-GRASP, to solve the p-median problem. This proposal was based on the hypothesis that patterns extracted from sub-optimal obtained solutions could guide the search for better ones. The experimental results showed that the proposed strategy was able to obtain better solutions in less computational time than the original GRASP.

In the first version of the hybrid GRASP, the data mining process occurred just once. In order to try to explore the gradual evolution of the elite set of solutions and allow the extraction of better and higher-quality patterns, we proposed another version of the hybrid strategy, called MDM-GRASP. This strategy extracts new sets of patterns whenever the elite set changes and become stable. The conducted experiments showed that the MDM-GRASP obtained even better results than the DM-GRASP, not only in terms of quality, but also regarding the computational time.

These encouraging results obtained with DM-GRASP and MDM-GRASP motivate us, as future work, to try to introduce into other metaheuristics the idea of extracting patterns from sub-optimal solutions and using them in search procedures. We believe that other metaheuristics and many combinatorial optimization problems can benefit from the incorporation of data mining techniques.

References

[1] R. Agrawal and R. Srikant, *Fast algorithms for mining association rules*, Proceedings of the Very Large Data Bases Conference, pp. 487-499, 1994.

[2] T. A. Feo and M. G. C. Resende, *A probabilistic heuristic for a computationally difficult set covering problem*, Operations Research Letters, 8 (1989), pp. 67-71.

[3] T. A. Feo and M. G. C. Resende, *Greedy randomized adaptive search procedures*, Journal of Global Optimization, 6 (1995), pp. 109-133.

[4] P. Festa and M. G. C. Resende, *GRASP: An annotated bibliography*, Essays and Surveys in Metaheuristics, Kluwer Academic Publishers, 2002.

[5] M. D. H. Gamal and S. Salhi, *A cellular heuristic for the multisource Weber problem*, Computers & Operations Research, 30 (2003), pp. 1609-1624.

[6] B. Goethals and M. J. Zaki, *Advances in Frequent Itemset Mining Implementations: Introduction to FIMI03*, Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations, 2003.

[7] G. Grahne and J. Zhu, *Efficiently using prefix-trees in mining frequent item-sets*, Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations, 2003.

[8] J. Han, J. Pei, and Y. Yin, *Mining frequent patterns without candidate generation*, Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 1-12, 2000.

[9] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2^{nd} Ed., Morgan Kaufmann Publishers, 2006.

[10] O. Kariv and L. Hakimi, *An algorithmic approach to network location problems, part ii: The p-medians*, SIAM Journal of Applied Mathematics, 37 (1979), pp. 539–560.

[11] N. Mladenović, J. Brimberg, P. Hansen and José A. Moreno-Pérez, *The p-median problem: A survey of metaheuristic approaches*, European Journal of Operational Research, 179 (2007), pp. 927-939.

[12] S. Orlando, P. Palmerimi, and R. Perego, *Adaptive and resource-aware mining of frequent sets*, Proceedings of the IEEE International Conference on Data Mining, pp. 338-345, 2002.

[13] I. Osman and G. Laporte, *Metaheuristics: A bibliography*, Annals of Operations Research, 63 (1996), pp. 513-623.

[14] M. G. C. Resende and C. C. Ribeiro, *Greedy randomized adaptive search procedures*, Handbook of Metaheuristics, Kluwer Academic Publishers, 2003.

[15] M. H. F. Ribeiro, V. F. Trindade, A. Plastino, and S. L. Martins, *Hybridization of GRASP metaheuristic with data mining techniques*, Proceedings of the ECAI Workshop on Hybrid Metaheuristics, pp. 69-78, 2004.

[16] M. H. F. Ribeiro, V. F. Trindade, A. Plastino, and S. L. Martins, *Hybridization of GRASP metaheuristic with data mining techniques*, Journal of Mathematical Modeling and Algorithms, 5 (2006), pp. 23-41.

[17] S. Salhi, *Heuristic Search: The Science of Tomorrow*, OR48 Keynote Papers, Operational Research Society, pp. 38–58, 2006.

[18] L. F. Santos, M. H. F. Ribeiro, A. Plastino, and S. L. Martins, *A hybrid GRASP with data mining for the maximum diversity problem*, Proceedings of the International Workshop on Hybrid Metaheuristics, LNCS 3636, pp. 116–127, 2005.

[19] L. F. Santos, C. V. Albuquerque, S. L. Martins, and A. Plastino, *A hybrid GRASP with data mining for efficient server replication for reliable multicast*, Proceedings of the IEEE GLOBECOM Conference, 2006.

[20] L. F. Santos, S. L. Martins, and A. Plastino, *Applications of the DM-GRASP heuristic: A survey*, International Transactions in Operational Research, 15 (2008), pp. 387–416.

[21] E. G. Talbi, *A taxonomy of hybrid metaheuristics*, Journal of Heuristics, 8(2002), pp. 541-564.

[22] B. C. Tansel, R. L. Francis, and T. J. Lowe. *Location on networks: A survey*, Management Science, 29 (1983), pp. 482–511.

[23] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques.* 2^{nd} Ed., Morgan Kaufmann Publishers, 2005.