Universidade Federal Fluminense
Instituto de Computação
Departamento de Ciência da Computação

Hugo Barbalho

A Hybrid Data Mining GRASP with Path-Relinking

Niterói-RJ
2011

Hugo Barbalho

A Hybrid Data Mining GRASP with Path-Relinking

Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal Fluminense como parte dos requisitos para obtenção do Grau de Bacharel em Ciência da Computação.

Orientadores:
Profa. Dr. Simone Martins, Prof. Dr. Alexandre Plastino e Profa. Dr. Isabel Rosseti

Niterói-RJ
2011

Hugo Barbalho

A Hybrid Data Mining GRASP with Path-Relinking

Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal Fluminense como parte dos requisitos para obtenção do Grau de Bacharel em Ciência da Computação.

Aprovada em junho de 2011.

BANCA EXAMINADORA

Profa. Simone Martins, D.Sc - Orientadora
UFF

Prof. Alexandre Plastino, D.Sc - Orientador
UFF

Profa. Isabel Rosseti, D.Sc - Orientadora
UFF

Prof. Yuri Frota, D.Sc
UFF

Prof. Sabir Ribas, M.Sc
UFF

Niterói-RJ

2011

Agradecimentos

À minha mãe, responsável por todos os meus sucessos, pois sempre acreditou e apoiou todos os meus planos.
Aos meus pais, por fornecerem uma sólida estrutura familiar.
Ao meu irmão, por sempre dar bons conselhos quando precisei tomar decisões importantes antes e durante toda a faculdade.
À minha irmã, por tornar essa jornada menos estressante.
Aos meus avós, por sempre apoiarem nossa família.
À minha namorada Cinthya, por estar ao meu lado em todos os momentos difíceis da faculdade, com muita paciência e carinho.
Ao Professor Alexandre Plastino pela orientação acadêmica, pelo convite para iniciação científica e por sua importante contribuição para o sucesso desse trabalho.
À Professora Simone Martins, por sua grande contribuição para criação e apresentação desse trabalho.
À Professora Isabel Rosseti, por sua importante colaboração e orientação nas tarefas de desenvolvimento desse trabalho.
Às Professoras Ana Farias e Dirce Uesu, pela orientação no meu primeiro projeto na faculdade.
Ao Professor Esteban Clua, por apoiar minha participação na Maratona de Programação.

List of Figures

List of Tables

Contents

Resumo

Metaheurísticas são utilizadas para resolver, de forma aproximada, problemas de otimização combinatória computacionalmente difíceis para os quais a solução ótima não pode ser encontrada garantidamente em um tempo computacionalmente viável. A metaheurística GRASP vem sendo amplamente aplicada em problemas de otimização como escalonamento e roteamento, e bons resultados em termos de qualidade e tempo computacional vêm sendo obtidos com sua utilização. A metaheurística GRASP é uma estratégia iterativa de fácil implementação que retorna a melhor solução encontrada dentre todas as iterações. Cada iteração é composta de duas fases: construção e busca local. Na primeira fase uma solução viável é construída e na fase seguinte sua vizinhança é explorada na tentativa de encontrar soluções melhores. A hibridização de metaheurísticas - combinação de metaheurísticas com conceitos e processos de outras áreas de pesquisa - vem sendo uma importante linha de pesquisa em otimização combinatória. Processos de mineração de dados são utilizados para extrair conhecimento, expresso por meio de regras ou padrões, de um conjunto de dados de forma automática. Uma versão híbrida da metaheurística GRASP, que incorpora um módulo de mineração de dados, foi desenvolvida e bons resultados foram obtidos com sua utilização. Nessa proposta, após executar um número significativo de iterações, um módulo de mineração de dados extrai padrões frequentes de um conjunto elite que contém soluções sub-ótimas geradas nas iterações. Esses padrões apresentam características de soluções de boa qualidade e podem ser utilizados para auxiliar as próximas iterações do GRASP na busca através do espaço de soluções. A hibridização da metaheurística GRASP com mineração de dados já foi aplicada em alguns problemas combinatórios, tais como: problema de empacotamento, problema de maximização da diversidade, problema do multicast confiável e o problema das p-medianas. Neste trabalho, pretende-se mostrar que tanto a metaheurística GRASP original como a metaheurística GRASP que incorpora uma técnica de reconexão por caminhos podem se beneficiar da mineração de dados. Para tanto, será utilizado o problema de síntese de redes a k-caminhos (kPNDP), com k igual a dois, que recentemente foi resolvido usando a metaheurística GRASP com reconexão de caminhos e foram obtidos excelentes resultados.

Experimentos computacionais, comparando a metaheurística GRASP com reconexão por caminhos e diferentes formas híbridas do GRASP com mineração de dados propostas neste trabalho, mostraram que a incorporação de um módulo de mineração de dados permitiu que a heurística híbrida encontrasse melhores resultados em menor tempo computacional. A heurística GRASP com reconexão de caminhos originalmente desenvolvida para este problema havia obtido os melhores resultados conhecidos em termos de qualidade. Portanto, outra importante contribuição deste trabalho foi melhorar a qualidade desses resultados.

Palavras-chave:
Metaheurística, Reconexão por caminhos, Mineração de dados, GRASP, Hibridização, 2PNDP.

Abstract

The exploration of hybrid metaheuristics – combination of metaheuristics with concepts and processes from other research areas – has been an important trend in combinatorial optimization research. An instance of this study is the hybrid version of the GRASP metaheuristic that incorporates a data mining process. 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 optimization 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. The hybrid data mining GRASP has been successfully applied for different combinatorial problems: the set packing problem, the maximum diversity problem, the server replication for reliable multicast problem and the p-median problem. In this work, we show that, not only the traditional GRASP, but also GRASP improved with the path-relinking heuristic – a memory-based intensification strategy – could benefit from exploring a data mining procedure. Computational experiments, comparing traditional GRASP with path-relinking and different path-relinking hybrid proposals, showed that employing the combination of path-relinking and data mining made the GRASP find better results in less computational time. Another contribution of this work is the application of the path-relinking hybrid proposal for the 2-path network design problem, which improved the state-of-the-art solutions for this problem.

Keywords:
GRASP, Path-Relinking, Data Mining, Hybrid Metaheuristic, 2-Path Network Design Problem.

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 (17). An overview of heuristic search can be found in (26).

A trend in metaheuristics research is the exploration of hybrid metaheuristics (30). One kind of such hybrid methods results from the combination of concepts and strategies behind two or more classic metaheuristics, and another kind corresponds to metaheuristics combined with concepts and processes from other research areas responsible for performing specific tasks that can improve the original method. An instance of the latter case is the hybrid version of the GRASP metaheuristic that incorporates a data mining process, called DM-GRASP (Data Mining GRASP) (29).

The GRASP (Greedy Randomized Adaptive Search Procedures) metaheuristic (4, 5), since it was proposed, has been successfully applied to solve many optimization problems, in several areas like scheduling, routing, partitioning, location and assignment (6, 7). GRASP is easy to implement and is able to obtain very good solutions in acceptable computational times (7).

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 (12, 31).

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 (21, 22). The basic hypothesis 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 ideas of keeping track of recurrent good sub-optimal solutions and fixing variables have been successfully explored coupled with other heuristics. Lin and Kernighan (14) developed a multistart heuristic for the travelling salesman problem, where they fix some links observed to occur in a number of previously locally optimum tours found by the algorithm. Lodi et al. (15) developed an evolutionary heuristic for quadratic 0-1 programming, where they present an intensification strategy used in a genetic algorithm to fix variables, which can have their values fixed during all steps of the algorithm or only during a given number of steps. Fleurent and Glover (8) described multistart strategies for the quadratic assignment problem, where, during the constructive procedure, they select elements to be inserted in a solution from an elite set containing the best solutions generated so far.

The aim of the hybrid data mining proposal is to use specific techniques found in the data mining research area to search for good patterns extracted from a set of high quality solutions. 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 three other applications, namely, the maximum diversity problem (27), the server replication for reliable multicast problem (28) and the p-median problem (18), and the results were equally successful.

The first contribution of this work is to show that not only the traditional GRASP metaheuristic but also GRASP procedures improved with the path-relinking heuristic – a memory-based intensification mechanism – can benefit from the incorporation of a data mining procedure to extract patterns of sub-optimal solutions in order to guide more efficiently the search for better solutions.

Path-relinking was originally proposed by Glover (9) as an intensification strategy exploring trajectories connecting elite solutions obtained by tabu search or scatter search strategies. Starting from one or more elite solutions, path-relinking generates paths leading toward other elite solutions and explores them in the search for better solutions. To generate paths, moves are selected to introduce attributes in the current solution that are present in the elite guiding solution. Path-relinking is a strategy that seeks to incorporate attributes of high quality solutions, by favoring them in the selected moves.

In this work, we present two path-relinking hybrid strategies, called DM-GRASP-PR and MDM-GRASP-PR, which combine a data mining procedure into the GRASP with path-relinking, and show that these strategies can improve the solution quality and computational time of the original GRASP with path-relinking.

The second contribution is the application of the path-relinking hybrid proposals to solve the 2-path network design problem (2PNDP). This problem has shown to be NP-hard and many applications of this problem can be found in the design of communication networks, in which paths with few edges are sought to enforce high reliability and small delays (23). Recent procedures developed to solve this problem are based on metaheuristics, and GRASP procedures with path-relinking have achieved excellent results (24). The computational experiments conducted in this work show that the implemented path-relinking hybrid strategies were able to improve the state-of-the-art solutions for the 2PNDP.

The remaining of this work is organized as follows. In Chapter 2, we review the main concepts and the structure of both GRASP metaheuristic and path-relinking strategy. In Chapter 3, we present the hybrid strategy DM-GRASP-PR developed for the 2PNDP and compare the computational results obtained by this strategy and the traditional GRASP with path-relinking. In Chapter 4, the strategy MDM-GRASP-PR is described and computational results are presented comparing the DM-GRASP-PR and the MDM-GRASP-PR strategies. Finally, in Chapter 5, concluding remarks are made and some future works are pointed out.

2 GRASP with path-relinking

GRASP (19) is a metaheuristic already applied successfully to many optimization problems (6, 7). 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   x* ← ∅;
2   repeat
3       x ← Construction_Phase();
4       x ← Local_Search_Phase(x);
5       if (f(x) > f(x*)) then
6           x* ← x;
7       end_if;
8   until TerminationCriterion();
9   return x*;
end.
```

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. In line 4, one component is randomly selected from this list and incorporated into the current solution in line 5. In line 7, the complete solution is returned.

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

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.

2.1 Path-relinking

Path-relinking is a technique proposed by Glover (9) to explore possible trajectories connecting high quality solutions obtained by heuristics like tabu search and scatter search.

The GRASP metaheuristic is a memoryless method, because all iterations are independent and no information about the solutions is passed from one iteration to another. The objective of introducing path-relinking to a pure GRASP algorithm is to retain previous good solutions and use them as guides in the search of new good solutions.

Laguna and Martí (13) were the first to use path-relinking within a GRASP strategy. Several extensions, improvements and successful applications of this technique can be found in the literature (20).

Basically, path-relinking is applied to a pair of solutions $\{s_i, s_g\}$ by starting from the initial solution s_i and gradually incorporating attributes from the guide solution s_g to s_i, until s_i becomes equal to s_g.

To use path-relinking within a GRASP procedure, an elite set P is maintained, in which good solutions found in previous GRASP iterations are stored.

Two basic strategies for introducing path-relinking into GRASP may be used (20): (a) performing path-relinking after each GRASP iteration using a solution from the elite set and a local optimum obtained after the GRASP local search, and (b) applying path-relinking to all pairs of elite solutions, either periodically or after all GRASP iterations terminate.

Path-relinking is performed between two solutions and there are several ways to explore the paths between them (20): backward relinking, forward relinking, backward-and-forward relinking, periodical relinking, randomized relinking and truncated relinking.

We show in Figure 3 the details of a path-relinking procedure specified for a minimization problem and using a single traverse from s_i to s_g. In lines 1 and 2, from the two solutions passed as parameters, x_1 and x_2, the initial and guide solutions are set. In line 3, the set Δ composed of positions in which s_i and s_g differ is calculated. The initial best solution and its cost are determined in lines 4 and 5. From lines 6 to 15, the steps of path-relinking are performed until the entire path from s_i to s_g is traversed. For every position $m \in \Delta$, let $s_i \oplus m$ be the solution obtained from s_i by changing its m-th position by that of s_g. In line 7 the component m^* of Δ for which $s_i \oplus m$ results in the least-cost solution is obtained. This component is removed from Δ in line 8 and the current solution is updated in line 9 by changing the value of its m^* position. This solution then is more similar to the guide solution because one element from the initial solution was replaced by another from the guide solution. In line 10 and 11, if this new intermediate solution has a better cost than the current best intermediate solution ($BestSolPR$), then the latter and its cost are updated. In line 14, the intermediate solution is set as the initial solution for the next step of the path-relinking.

```
procedure Path_Relinking(x_1, x_2);
    s_i ← SetSolIni(x_1, x_2);
    s_g ← SetSolGuide(x_1, x_2);
    Δ ← Compute_Difference(s_i, s_g);
    BestCostSolPR ← min{f(s_i), f(s_g)};
    BestSolPR ← argmin{f(s_i), f(s_g)};
    while |Δ| > 1 do
        m* ← argmin{f(s_i ⊕ m) : m ∈ Δ};
        Δ ← Δ \ m*;
        InterSol ← s_i ⊕ m*;
        if f(InterSol) > BestCostSolPR then
            BestCostSolPR ← f(InterSol);
            BestSolPR ← InterSol;
        end_if;
        s_i ← InterSol;
    end_while;
    return BestSolPR;
end.
```

Figure 3: Pseudo-code for path-relinking

3 The Hybrid DM-GRASP-PR Proposal

In this chapter, we describe the 2-path network design problem and the GRASP with path-relinking procedure developed in (24) to solve this problem. Then we present the DM-GRASP-PR heuristic, which is a hybrid version of the GRASP metaheuristic with path-relinking presented in (24) incorporated with a data mining process.

3.1 2-path network design problem

Let $G = (V, E)$ be a connected undirected graph, where V is the set of nodes and E is the set of edges. A k-path between nodes $s, t \in V$ is a sequence of at most k edges connecting them. Given a non-negative weight function $w : E \rightarrow R_+$ associated with the edges of G and a set D of pairs of origin-destination nodes, the *2-path network design problem* (2PNDP) consists in finding a minimum weighted subset of edges $E\prime \subseteq E$ containing a 2-path between every origin-destination pair in D. Applications of the 2PNDP can be found in the design of communication networks, in which paths with few edges are sought to enforce high reliability and small delays. The decision version of the 2PNDP has been proved to be NP-complete by Dahl and Johannessen (3). In (24), the authors successfully applied GRASP with path-relinking heuristics for approximately solving this problem.

3.2 GRASP-PR for 2PNDP

In this section, we present the GRASP heuristic with path-relinking (GRASP-PR) for the 2-path network design problem presented in (24).

3.2.1 Construction phase

The greedy randomized construction algorithm computes one shortest 2-path at-a-time. To avoid that edge weights be considered more than once, the weights of the edges already included in the solution are temporally made equal to zero during the forthcoming computations.

Figure 4 illustrates the pseudo-code of the construction phase of the GRASP with path-relinking heuristic for the 2PNDP. Initializations are performed in lines 1 and 2. Solution x is computed from scratch using edge weights w' that are initially equal to the original weights w. The loop in lines 3 to 9 is performed until a 2-path has been computed for every origin-destination pair. Each iteration starts by the random selection in line 4 of a pair (a, b) still to be routed. A shortest path P from a to b using the modified weights w' is computed in line 5. The weights of the edges in P are temporarily set to 0 for the remaining iterations in line 6. Pair (a, b) is removed in line 7 from the set of origin-destination pairs to be routed and in line 8 the edges in P are inserted into the solution under construction.

Since the loop is executed $|D|$ times and each shortest 2-path can be computed in time $O(|V|)$, the complexity of the construction procedure is $O(|V| \cdot |D|)$.

procedure `GreedyRandomizedConstruction2Path`($Seed$);
1 $x \leftarrow \emptyset$;
2 $w' \leftarrow w$;
3 **while** $D \neq \emptyset$ **do**
4   Select at random an yet unrouted origin-destination pair $(a, b) \in D$;
5   Compute the shortest 2-path P from a to b using weights w';
6     $w'_{ij} \leftarrow 0$ for all edges (i, j) in P;
7     $D \leftarrow D \setminus (\{a, b\})$;
8     $x \leftarrow x \cup P$;
9 **end_while**;
10 **return** x;
end.

Figure 4: Pseudo-code of the construction phase of the GRASP for the 2PNDP

3.2.2 Local search

Each solution x may be viewed as a collection of $|D|$ 2-paths. Given any solution x, its neighbor solutions $x\prime$ may be obtained by replacing any 2-path in x by another 2-path between the same origin-destination pair. The local search phase attempts to improve the solutions built greedily during the construction phase.

Figure 5 summarizes the pseudo-code of the local search procedure for the 2PNDP. The neighbor solution $x\prime$ and the modified edge weights are initialized respectively in lines 1 and 2. Variable *nochanges* initialized in line 3 is used as a flag to indicate that a local optimum has been found. A circular permutation of the demand pairs in D is created at random in line 4. The loop in lines 5 to 16 is performed until all $|D|$ 2-paths in the current solution have been consecutively examined and no shorter 2-path has been found, indicating that the current solution is locally optimal. Each iteration starts in line 6 by considering the next origin-destination pair (a, b), according with the circular permutation computed in line 4 and attempting to improve this 2-path. The following steps are performed: temporarily reset to zero the modified weights $w\prime$ of all edges used by the other 2-paths (line 7); compute the shortest 2-path from a to b using the modified edge weights $w\prime$ (line 8); update the incumbent solution $x\prime$ if the weight of the new 2-path is shorter (lines 9 and 10); and update the number of consecutive 2-paths examined without change in the current solution (lines 11 and 13). Once the iteration is finished, all weights are reset to their original values w in line 15. If less than $|D|$ 2-paths have been consecutively examined without improvement in the current solution, then a new iteration resumes. Otherwise, the eventually modified neighbor solution $x\prime$ is returned in line 17. The complexity of each iteration of local search is $O(n)$.

3.2.3 Path-relinking

The algorithm in Figure 6 illustrates the GRASP with path-relinking procedure for the 2PNDP. Since each solution to 2PNDP is characterized by a set of $|D|$ 2-paths between the extremities of every origin-destination pair, the symmetric difference $\Delta(x, x_t)$ between the current solution x and the target solution x_t can be easily computed and amounts to the set of 2-paths that appear in one of them but not in the other. Each move $m \in \Delta(x, x_t)$ is defined by one 2-path to be removed from and another to be inserted into the current solution x.

```
procedure LocalSearch2Path;
x′ ← x;
w′ ← w;
nochanges ← 0;
Create a random circular permutation of the demand pairs in D;
while nochanges < |D| do
    Select the next origin-destination pair (a, b);
    Temporarily reset to 0 the weights w′ of all edges appearing
    in the 2-paths connecting the remaining origin-destination pairs in D;
    Compute the shortest 2-path from a to b using the modified weights w′;
    if the weight of the new 2-path is smaller then
        Update solution x′ by using the new 2-path;
        nochanges ← 0;
    else
        nochanges ← nochanges + 1;
    end_if;
    Reset the weights w of all edge weights temporarily set to 0;
end_while;
return x′;
end.
```

Figure 5: Pseudo-code of the local search phase of the GRASP for the 2PNDP

Each GRASP iteration has now three main steps:

- *Construction phase:* procedure `GreedyRandomizedConstruction2Path` is used to build a feasible solution;
- *Local search phase:* procedure `LocalSearch2Path` is applied to the solution built in the construction phase and a local minimum is found; and
- *Path-relinking phase:* procedure `PathRelinking` is applied to the solution obtained by local search and to a randomly selected solution from the pool P twice (one using the latter as the starting solution and the other using the former). The locally optimal solution obtained by local search and the best solutions found along each relinking trajectory are considered as candidates for insertion into the pool. A solution is inserted in the pool if it is different from all solutions of the pool and its cost is better than the cost of the worst solution of the pool.

```
procedure GRASPwithPR2Path(MaxIterations, Seed)
    P ← ∅;
    f* ← ∞;
    for k = 1, . . . , MaxIterations do
        x ← GreedyRandomizedConstruction2Path(Seed);
        x ← LocalSearch2Path(x);
        Update the pool of elite solutions P with x;
        if |P| ≥ 2 then
            Select at random an elite solution y from the pool P;
            x1 ← PathRelinking(x, y);
            Update the pool of elite solutions P with x1;
            x2 ← PathRelinking(y, x);
            Update the pool of elite solutions P with x2;
            Set x ← argmin{f(x), f(x1), f(x2)};
        end_if;
        if f(x) < f* then
            x* ← x;
            f* ← f(x);
        end_if;
    end_for;
    return x*;
end.
```

Figure 6: GRASP with path-relinking for the 2PNDP

3.3 DM-GRASP-PR heuristic

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.

We have already developed heuristics hybridizing GRASP with data mining procedures, called DM-GRASP procedures, which were able to improve the quality of solutions in reasonable computational time for many problems like the set packing problem, the maximum diversity problem, the server replications for reliable multicast problem, and the p-median problem (18, 27–29).

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 compose the elite set.

After this first phase, the data mining process is applied to extract 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 (12). 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.

We developed the hybrid procedure DM-GRASP-PR, which incorporates a data mining procedure to a GRASP with path-relinking heuristic, in order to show that not only the traditional GRASP metaheuristic but also GRASP procedures improved with the path-relinking heuristic – a memory-based intensification mechanism – can benefit from the incorporation of a data mining procedure to extract patterns of sub-optimal solutions and guide the search for better solutions.

The pseudo-code of the DM-GRASP-PR for the 2PNDP is illustrated in Figure 7. In line 3, the elite set used for data mining is initialized with the empty set. The loop from line 4 to line 21 corresponds to the elite set generation phase, in which GRASP with path-relinking is performed for n iterations. The original construction method is executed in line 5, followed by the local search method in line 6 and the path-relinking procedure executed from line 8 to line 15. The elite set M, composed of d solutions, is updated in line 16. A solution is inserted in the elite set if it is not already in the set and its cost is better than the worst cost found in the set. In line 18, the best solution is updated, if the new generated solution presents a better cost than the best solution found in previous iterations. In line 22, the data mining procedure extracts t patterns from the elite set, which are inserted in decreasing order of pattern size in the set of patterns. The loop from line 23 to line 38 corresponds to the hybrid phase. In line 24, one pattern is picked from the set of patterns in a round-robin way. Then, the adapted construction procedure is performed in line 25, using the selected pattern. In line 26, the local search is executed. From line 28 to 33 the path-relinking procedure is executed. If a better solution is found, the best solution is updated in line 35. After the execution of all iterations, the best solution is returned in line 39.

The extraction of patterns from the elite set, which is activated in line 22 of the pseudo-code presented in Figure 7, 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 is a subset of I and a dataset T 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 T. The FIM problem consists of extracting all frequent itemset from a dataset T with a minimum support specified as a parameter. During the last two decades, many algorithms have been proposed to efficiently mine frequent itemsets (1, 10, 11, 16).

In this work, the useful patterns to be mined are sets of edges that commonly appear in sub-optimal solutions of the 2PNDP. This is a typical frequent itemset mining application, where the set of items is the set of potential edges. 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 edges 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-PR proposal for the 2PNDP, we decided to extract only maximal frequent itemset.

In Figure 8, the pseudo-code of the adapted construction is illustrated. It is quite similar to the code described in Figure 4 with the difference that, in line 5, we try to construct a 2-path between a pair (a, b) using only the edges from the pattern or the edges already used which had their weight modified to 0. If a 2-path was not found using just these edges, in line 7, we compute a 2-path starting from the partial solution found so far and using all edges from E.

3.4 Computational Results for DM-GRASP-PR

In this section, the computational results obtained for GRASP-PR and DM-GRASP-PR strategies are presented and compared. We generated 25 instances similar to the instances generated in (24). The instances are complete graphs with $|V| \in \{100, 200, 300, 400, 500\}$. The edge costs were randomly generated from the uniform distribution on the interval (0, 10] and $10 \times |V|$ origin-destination pairs were randomly chosen.

The algorithms were implemented in C and compiled with gcc 4.4.1. 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-PR and DM-GRASP-PR were run 10 times with a different random seed in each run. Each strategy executed a total of 1000 iterations.

After having conducted some tuning experiments, we set some parameter values. The size of the elite set (d), from which the patterns are mined, and the size of the set of patterns (t) were set to 10. And a set of edges was considered a pattern if it was present in at least two of the elite solutions.

In Table 1, the results related to the quality of the obtained solutions are shown. The first column presents the identifier of the instance ax-y, where $x = |V|$ and y is the seed used to generate the random instance parameters. The second and fourth columns present the best cost values obtained by GRASP with path-relinking (GRASP-PR) and the path-relinking hybrid proposal DM-GRASP-PR, and the third and fifth columns present the average cost values obtained by them. The smallest values, i.e., the better results, are bold-faced.

These results show that the proposed DM-GRASP-PR strategy was able to improve all results obtained by GRASP with path-relinking.

Table 2 presents the results related to execution time of both strategies. In these tables, the first column presents the instance identifier, the second and third columns show the average execution time (in seconds) of GRASP-PR and DM-GRASP-PR, obtained for 10 runs. The fourth column shows the percentual difference between the GRASP-PR and DM-GRASP-PR average times in relation to the GRASP average time.

For all instances, the execution times for DM-GRASP-PR were smaller than those for GRASP-PR. The last line of the table presents the average of the percentual differences. We can observe that, on average, DM-GRASP-PR was 20.23% faster than GRASP-PR.

There are two main reasons for the faster behavior of DM-GRASP-PR. First, the computational effort of the adapted construction phase is smaller than the original construction, since a smaller set of edges is processed to find a 2-path for each pair. In the adapted construction of the hybrid procedure, in a first attempt (line 5, Figure 8), only the edges from the pattern and edges with cost 0 are examined to construct a 2-path for a demand pair. 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.

	GRASP-PR		DM-GRASP-PR	
Instance	Best	Avg	Best	Avg
a100-1	679	687.5	**676**	**682.0**
a100-10	663	669.8	**662**	**668.7**
a100-100	670	674.6	**666**	**670.3**
a100-1000	644	649.9	**641**	**647.0**
a100-10000	664	669.2	**661**	**666.5**
a200-1	1386	1391.9	**1379**	**1384.6**
a200-10	1374	1386.0	**1362**	**1376.1**
a200-100	1361	1369.4	**1354**	**1362.0**
a200-1000	1363	1374.5	**1358**	**1367.9**
a200-10000	1375	1387.4	**1369**	**1377.5**
a300-1	2106	2117.0	**2081**	**2102.4**
a300-10	2134	2148.0	**2122**	**2133.7**
a300-100	2088	2096.2	**2072**	**2082.3**
a300-1000	2100	2105.7	**2080**	**2094.5**
a300-10000	2077	2092.8	**2067**	**2078.2**
a400-1	2807	2816.2	**2788**	**2797.5**
a400-10	2848	2864.7	**2833**	**2847.8**
a400-100	2818	2834.2	**2803**	**2818.9**
a400-1000	2822	2833.4	**2800**	**2816.4**
a400-10000	2856	2874.8	**2844**	**2857.2**
a500-1	3598	3606.6	**3571**	**3579.6**
a500-10	3595	3607.7	**3573**	**3580.7**
a500-100	3598	3612.4	**3576**	**3584.7**
a500-1000	3573	3592.0	**3554**	**3564.2**
a500-10000	3605	3625.0	**3580**	**3597.9**

Table 1: GRASP-PR and DM-GRASP-PR quality results

Instance	GRASP-PR	DM-GRASP-PR	%
a100-1	44.22	**37.39**	15.44
a100-10	43.29	**36.14**	16.51
a100-100	46.66	**38.89**	16.66
a100-1000	42.98	**36.11**	15.99
a100-10000	43.57	**36.87**	15.37
a200-1	201.30	**161.87**	19.59
a200-10	206.32	**166.02**	19.53
a200-100	197.35	**157.37**	20.26
a200-1000	199.61	**158.63**	20.53
a200-10000	207.02	**166.49**	19.58
a300-1	516.63	**401.89**	22.21
a300-10	515.14	**401.34**	22.09
a300-100	517.84	**412.27**	20.39
a300-1000	516.14	**398.99**	22.70
a300-10000	515.48	**399.88**	22.43
a400-1	1000.79	**769.70**	23.09
a400-10	1003.74	**780.44**	22.25
a400-100	1026.18	**854.99**	16.68
a400-1000	1022.98	**824.99**	19.35
a400-10000	1028.98	**808.78**	21.40
a500-1	1727.36	**1330.40**	22.98
a500-10	1712.67	**1302.53**	23.95
a500-100	1747.14	**1396.26**	20.08
a500-1000	1721.36	**1332.65**	22.58
a500-10000	1760.41	**1337.25**	24.04
Average			20.23

Table 2: GRASP-PR and DM-GRASP-PR time results

```
procedure DMGRASPPR2Path(MaxIterations, Seed, n, d, t)
P ← ∅;
f* ← ∞;
M ← ∅;
for k = 1, ..., n do
    x ← GreedyRandomizedConstruction2Path(Seed);
    x ← LocalSearch2Path(x);
    Update the pool of elite solutions P with x;
    if |P| ≥ 2 then
        Select at random an elite solution y from the pool P;
        x_1 ← PathRelinking(x, y);
        Update the pool of elite solutions P with x_1;
        x_2 ← PathRelinking(y, x);
        Update the pool of elite solutions P with x_2;
        Set x ← argmin{f(x), f(x_1), f(x_2)};
    end_if;
    UpdateElite(M, x, d);
    if f(x) < f* then
        x* ← x;
        f* ← f(x);
    end_if;
end_for;
patterns_set ← Mine(M, t);
for k = 1, ..., n do
    pattern ← SelectNextLargestPattern(patterns_set);
    x ← AdaptedGreedyRandomizedConstruction2Path(pattern);
    x ← LocalSearch2Path(x);
    Update the pool of elite solutions P with x;
    Select at random an elite solution y from the pool P;
    x_1 ← PathRelinking(x, y);
    Update the pool of elite solutions P with x_1;
    x_2 ← PathRelinking(y, x);
    Update the pool of elite solutions P with x_2;
    Set x ← argmin{f(x), f(x_1), f(x_2)};
    if f(x) < f* then
        x* ← x;
        f* ← f(x);
    end_if
end_for;
return x*;
end.
```

Figure 7: Hybrid GRASP with path-relinking for the 2PNDP

```
procedure AdaptedGreedyRandomizedConstruction2Path(Seed, pattern);
  x ← ∅;
  w′ ← w;
  while D ≠ ∅ do
      Select at random an yet unrouted origin-destination pair (a, b) ∈ D;
      Compute the shortest 2-path P from a to b using edges from pattern
      or edges with weight 0;
      if it was not possible to find a complete 2-path P then
          Compute the shortest 2-path P from a to b using the partial solution
          and the other edges not yet used;
      end_if
      w′_ij ← 0 for all edges (i, j) in P;
      D ← D \ ({a, b});
      x ← x ∪ P;
  end_while;
  return x;
end.
```

Figure 8: Pseudo-code of the adapted construction phase of the DM-GRASP-PR for the 2PNDP

4 The hybrid MDM-GRASP-PR proposal

In the proposed hybrid DM-GRASP-PR, 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 2PNDP, called MDM-GRASP-PR (Multi Data Mining GRASP-PR).

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 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-PR for the 2PNDP is illustrated in Figure 9. The loop from line 2 to 18 corresponds to the first elite set generation phase, in which GRASP iterations with path-relinking are performed until the elite set becomes ready to be mined or the termination criterion – the total number of iterations – becomes true. Next, in the loop from line 19 to 37, whenever the elite set is ready, the data mining procedure is executed in line 21. In line 23, 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 24, using the selected pattern as described in the previous chapter. In line 25, the local search is executed. From line 27 to 32 the path relinking procedure is executed. If a better solution is found, the best solution is updated in line 35. After the execution of all iterations, the best solution is returned in line 38.

```
procedure MDMGRASPPR2Path(MaxIterations, Seed, d, t)
P ← ∅; f* ← ∞; M ← ∅;
repeat
    x ← GreedyRandomizedConstruction2Path(Seed);
    x ← LocalSearch2Path(x);
    Update the pool of elite solutions P with x;
    if |P| ≥ 2 then
        Select at random an elite solution y from the pool P;
        x1 ← PathRelinking(x, y);
        Update the pool of elite solutions P with x1;
        x2 ← PathRelinking(y, x);
        Update the pool of elite solutions P with x2;
        Set x ← argmin{f(x), f(x1), f(x2)};
    end_if;
    UpdateElite(M, x, d);
    if f(x) < f* then
        x* ← x; f* ← f(x);
    end_if;
until elite_set_is_ready or end_criterion;
while not end_criterion
    if elite_set_is_ready then
        patterns_set ← Mine(M, t);
    end_if;
    pattern ← SelectNextLargestPattern(patterns_set);
    x ← AdaptedGreedyRandomizedConstruction2Path(pattern);
    x ← LocalSearch2Path(x);
    Update the pool of elite solutions P with x;
    Select at random an elite solution y from the pool P;
    x1 ← PathRelinking(x, y);
    Update the pool of elite solutions P with x1;
    x2 ← PathRelinking(y, x);
    Update the pool of elite solutions P with x2;
    Set x ← argmin{f(x), f(x1), f(x2)};
    UpdateElite(M, x, d);
    if f(x)) < f* then
        x* ← x; f* ← f(x);
    end_if;
end_while;
return x*;
end.
```

Figure 9: Hybrid MDM-GRASP-PR with path-relinking for the 2PNDP

4.1 Computational Results

In this section, we report the computational results obtained for the proposed MDM-GRASP-PR strategy. The 2PNDP instances are the same used in the previous chapter. The MDM-GRASP-PR was also run 10 times with a different random seed in each run. The number of executed iterations were 1000, the same used in the previous experiments. We performed some experiments using three values for the parameter used to define if the elite set is stable: 1%, 3% and 5% of the total number of iterations. We adopted 1% as this value provided the best cost values.

Since, in the previous analysis, the DM-GRASP-PR outperformed GRASP-PR, we decided to compare the MDM-GRASP-PR only with the DM-GRASP-PR strategy. In Table 3, the results related to quality of the obtained solutions are shown. MDM-GRASP-PR found 18 better results for best values and DM-GRASP-PR found four. MDM-GRASP-PR found 24 better results for average values and DM-GRASP-PR just one. These results show that the MDM-GRASP-PR proposal was able to improve the results obtained by DM-GRASP-PR.

Table 4 compares the execution times spent by DM-GRASP-PR and MDM-GRASP-PR. We can see that the DM-GRASP-PR was faster than the MDM-GRASP-PR in 18 instances and MDM-GRASP-PR was faster than DM-GRASP-PR in seven instances. However, we observe that MDM-GRASP-PR was, on average, just 1.34% slower than DM-GRASP-PR which is not very significant in terms of the heuristic performance. We conclude that both path-relinking hybrid proposals had a similar behavior in terms of computational time.

In order to verify whether or not the differences of mean values obtained by the evaluated strategies presented in Tables 1 and 3 are statistically significant, we employed the unpaired Student's t-test technique. Table 5 presents, for each pair of heuristics and for each instance group composed by instances with the same number of nodes, the number of better average solutions found by each strategy 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.

When comparing both DM-GRASP-PR and MDM-GRASP-PR with GRASP-PR (in the first four lines), we can note that almost all differences of performance are statistically

	DM-GRASP-PR		MDM-GRASP-PR	
Instance	Best	Avg	Best	Avg
a100-1	676	682.0	**674**	**681.9**
a100-10	662	668.7	**659**	**665.2**
a100-100	**666**	670.3	667	**670.0**
a100-1000	641	647.0	**640**	**646.7**
a100-10000	661	666.5	**658**	**665.4**
a200-1	**1379**	1384.6	1380	**1383.9**
a200-10	1362	1376.1	1362	**1372.5**
a200-100	1354	1362.0	**1352**	**1360.7**
a200-1000	1358	1367.9	**1356**	**1364.0**
a200-10000	1369	1377.5	**1363**	**1374.3**
a300-1	**2081**	2102.4	2082	**2099.3**
a300-10	**2122**	2133.7	2125	**2132.1**
a300-100	2072	2082.3	**2069**	**2076.3**
a300-1000	2080	2094.5	**2076**	**2090.3**
a300-10000	2067	2078.2	**2060**	**2075.1**
a400-1	2788	2797.5	**2786**	**2791.4**
a400-10	2833	2847.8	**2819**	**2844.1**
a400-100	2803	2818.9	2803	**2808.9**
a400-1000	2800	2816.4	**2793**	**2810.9**
a400-10000	2844	2857.2	**2793**	**2810.9**
a500-1	3571	3579.6	**3567**	**3576.9**
a500-10	3573	3580.7	**3566**	**3580.1**
a500-100	3576	3584.7	**3572**	**3583.1**
a500-1000	3554	**3564.2**	3554	3564.9
a500-10000	3580	3597.9	**3573**	**3596.1**

Table 3: DM-GRASP-PR and MDM-GRASP-PR quality results

significant. The last two lines represent the comparison between DM-GRASP-PR and MDM-GRASP-PR. In this comparison, we observe that MDM-GRASP-PR obtained, for all groups of instances, a greater number of better results. However, the difference of performance between DM-GRASP-PR and MDM-GRASP-PR was not statistically significant, considering a p-value less than 0.01. These results show the superiority of the data mining strategies, mainly the good behavior of the MDM-GRASP-PR.

Figures 10 to 12 illustrate the behavior of the construction, local search and path-relinking phases, in terms of the cost values obtained, by GRASP-PR, DM-GRASP-PR, and MDM-GRASP-PR throughout the execution of 1000 iterations, for the a400-100 instance, with a specific random seed.

The 2PNDP is a minimization problem and the figures show that the local search

Instance	DM-GRASP-PR	MDM-GRASP-PR	%
a100-1	**37.39**	38.50	-2.96
a100-10	**36.14**	37.54	-3.87
a100-100	**38.89**	40.41	-3.91
a100-1000	**36.11**	37.51	-3.89
a100-10000	**36.87**	38.41	-4.19
a200-1	**161.87**	163.19	-0.81
a200-10	**166.02**	167.06	-0.63
a200-100	**157.37**	162.58	-3.31
a200-1000	**158.63**	160.25	-1.02
a200-10000	**166.49**	166.61	-0.07
a300-1	**401.89**	409.38	-1.86
a300-10	**401.34**	410.17	-2.20
a300-100	412.27	**404.22**	1.95
a300-1000	398.99	**395.88**	0.78
a300-10000	**399.88**	403.97	-1.02
a400-1	769.70	**749.77**	2.59
a400-10	**780.44**	811.97	-4.04
a400-100	854.99	**799.67**	6.47
a400-1000	824.99	**797.91**	3.28
a400-10000	808.78	**797.91**	1.34
a500-1	**1330.40**	1349.39	-1.43
a500-10	**1302.53**	1346.80	-3.40
a500-100	**1396.26**	1413.63	-1.24
a500-1000	**1332.65**	1382.99	-3.78
a500-10000	**1337.25**	1420.02	-6.19
Average			-1.34

Table 4: DM-GRASP-PR and MDM-GRASP-PR time results

Strategy	Instance Group				
	a100	a200	a300	a400	a500
GRASP-PR	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)
DM-GRASP-PR	5 (2)	5 (2)	5 (5)	5 (5)	5 (5)
GRASP-PR	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)
MDM-GRASP-PR	5 (2)	5 (4)	5 (5)	5 (5)	5 (5)
DM-GRASP-PR	0 (0)	0 (0)	0 (0)	0 (0)	1 (0)
MDM-GRASP-PR	5 (0)	5 (0)	5 (0)	5 (0)	4 (0)

Table 5: Analysis of statistical significance

always reduces the cost of the solution obtained by the construction phase. We can also observe that the path-relinking procedure also always reduces the cost obtained after the local search.

In Figure 10, we observe that the construction and local search of GRASP-PR presents similar behavior throughout the iterations. The path-relinking procedure becomes more effective in reducing the cost, after some iterations, when the pool contains more solutions of better quality. In the last iterations, the path-relinking still improves the solution cost but with a smaller rate of improvement, because the pool contains less diverse solutions. The total time of this GRASP execution was 943.55s.

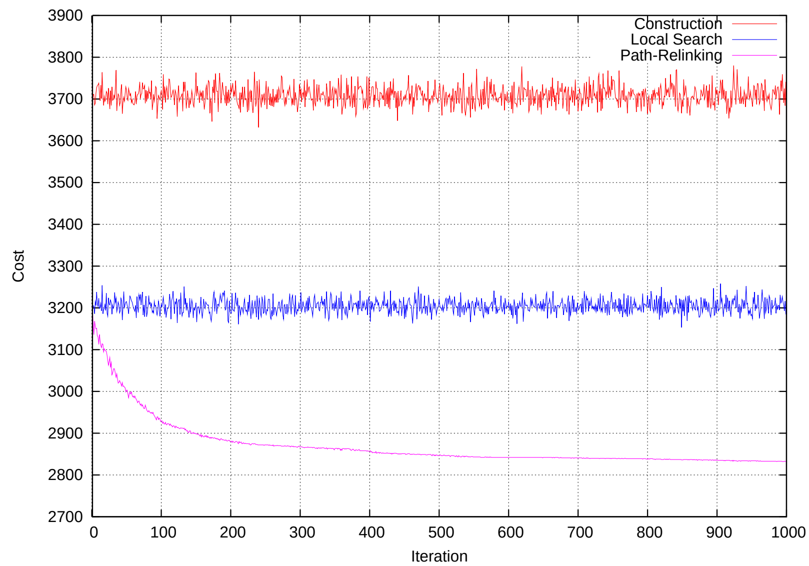

Figure 10: Cost × iteration plot of one execution of GRASP-PR for instance a400-100

In the DM-GRASP-PR strategy, the data mining procedure is executed immediately after iteration 500. We can observe, in Figure 11, that, from this point, the quality of the solutions obtained by the construction, local search and path-relinking phases are improved. The total time of this DM-GRASP-PR execution was 779.81s. The elite set generation phase took 487.17s, the data mining procedure 2.4s and the hybrid phase took 290.23s. This indicates that the data mining executing time is negligible when compared with time related to the DM-GRASP-PR iterations.

The behavior of MDM-GRASP-PR is presented in Figure 12. The data mining procedure was activated four times, after the iterations 584, 603, 654 and 822. We can observe that, in this specific execution, the improvement due to the activation of the data mining process started to happen immediately after the iteration 582, later than the mining execution by the DM-GRASP-PR. On the other hand, differently from the DM-GRASP-PR, we can observe that the MDM-GRASP-PR, after the first mining execution, continues to slightly and gradually reduce the cost of the solutions obtained by the construction, local

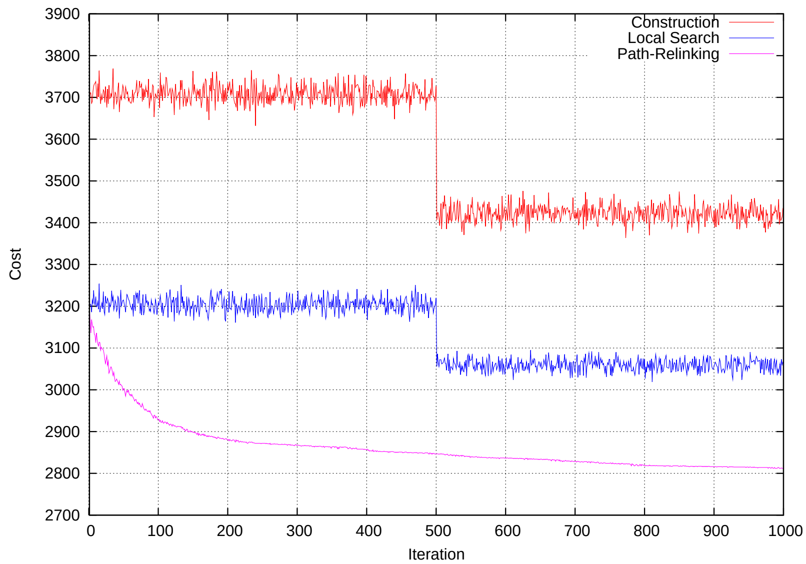

Figure 11: Cost × iteration plot of one execution of DM-GRASP-PR for instance a400-100

search and path-relinking phases, since patterns are extracted more than once.

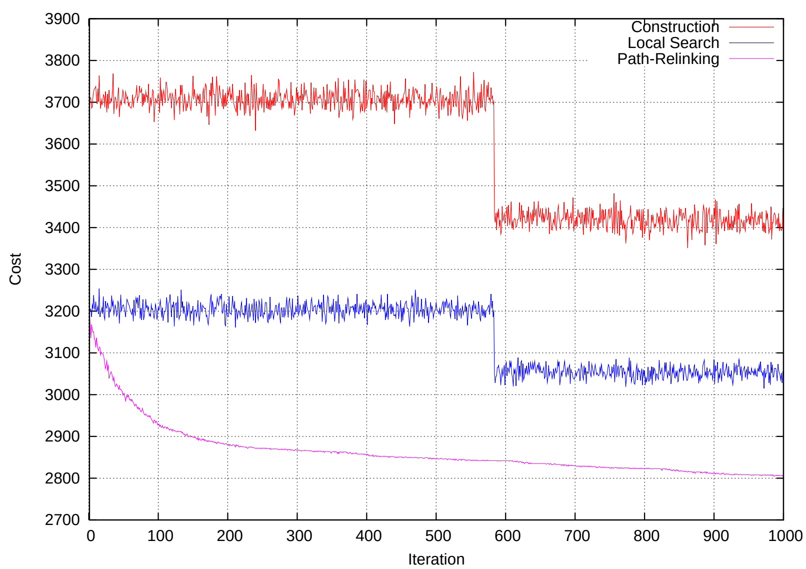

Figure 12: Cost × iteration plot of one execution of MDM-GRASP-PR for instance a400-100

The total time of this MDM-GRASP-PR execution was 830.62s. The elite set generation phase took 587.52s, the total time of the four data mining executions was 9.66s and the hybrid iterations took 243.11s. Again, we can observe that the sum of the time of all data mining activations is not relevant.

Figures 13 to 15 illustrate the behavior of the construction, local search and path-relinking phases, in terms of the computational times used by GRASP-PR, DM-GRASP-PR, and MDM-GRASP-PR throughout the same three executions of the 1000 iterations for the instance a400-100.

The figures show that for all strategies the path-relinking took more time than the local search which spends more time than the construction phase.

In Figure 13, we observe that the computational time spent by the construction, local search and path-relinking procedures of GRASP-PR are the same throughout the iterations.

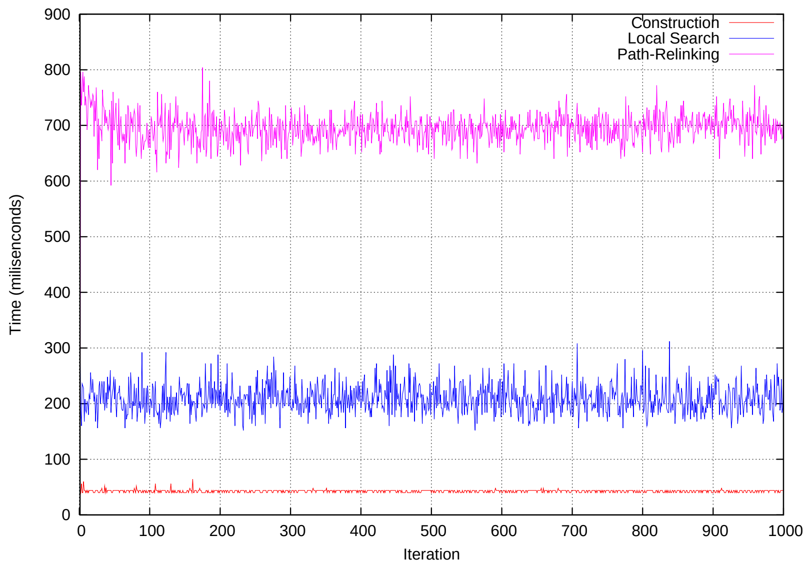

Figure 13: Time × iteration plot of one execution of GRASP-PR for instance a400-100

Since, in the DM-GRASP-PR strategy, the data mining procedure is executed after iteration 500, we can observe, in Figure 14, that, from this point, the computational times spent by the construction phase, the local search execution and, mainly, the path-relinking procedure are reduced. The construction phase spent less time because to find a 2-path for a pair origin-destination, in a first attempt, only the edges from the pattern and the edges with weight equal to 0 are examined. The solutions generated in the hybrid construction phase present better cost, so the local search took less time to find a local optima. The path-relinking is performed between a solution obtained after the local search and a randomly chosen solution from the pool. As the solutions generated after the local search procedure present better cost in the hybrid iterations, they are more similar to the solutions in the pool and the path-relinking procedure took less time to execute.

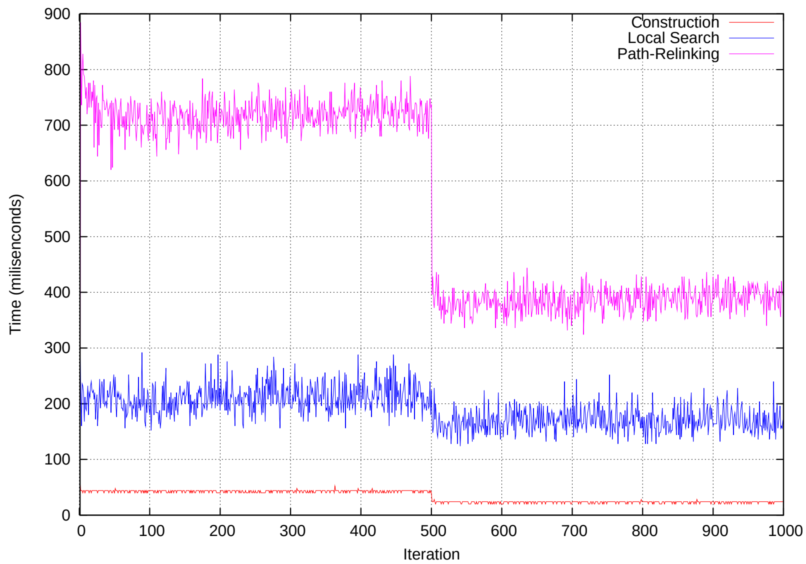

Figure 14: Time × iteration plot of one execution of DM-GRASP-PR for instance a400-100

In the MDM-GRASP-PR strategy, the data mining procedure is executed more than once. We can observe, in Figure 15, that MDM-GRASP-PR behaves similar to the DM-GRASP-PR until the first mining is performed. Then as more mining steps are executed the computational times gradually and slightly reduce for the construction, local search and path-relinking procedures.

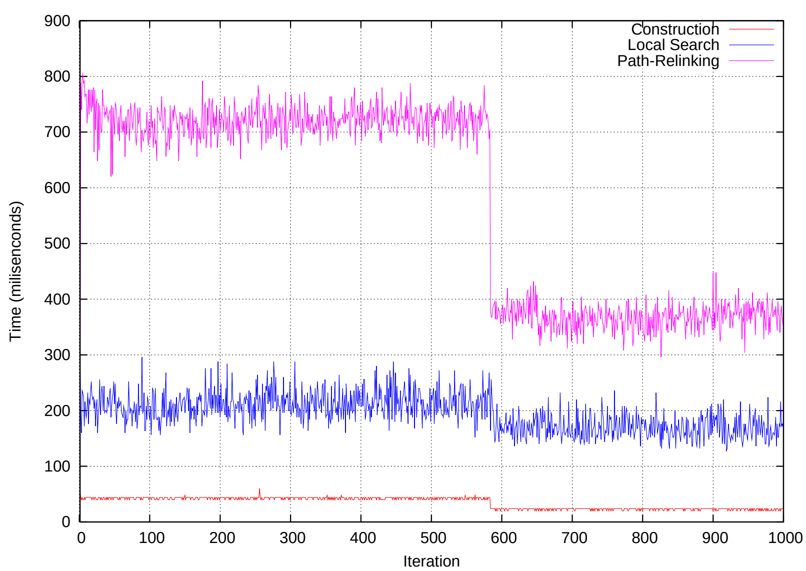

Figure 15: Time × iteration plot of one execution of MDM-GRASP-PR for instance a400-100

Figure 16 and 17 illustrate another execution – with a different random seed – of the

MDM-GRASP-PR for the same instance a400-100. Figure 16 shows the cost per iteration and Figure 17 presents the time per iteration plots. We can observe that, in this run, the first data mining execution was performed after iteration 361, before the mining performed in the other execution of the MDM-GRASP-PR (in the iteration 584). It means that, in this run, the elite set became stable earlier and the strategy could start using patterns soon after the first mining activation. Due to this anticipation, the reduction of the time spent by construction, local search and path-relinking phases started earlier and the total time of this MDM-GRASP-PR execution was 732.41s, faster than the other run, which took 830.62s.

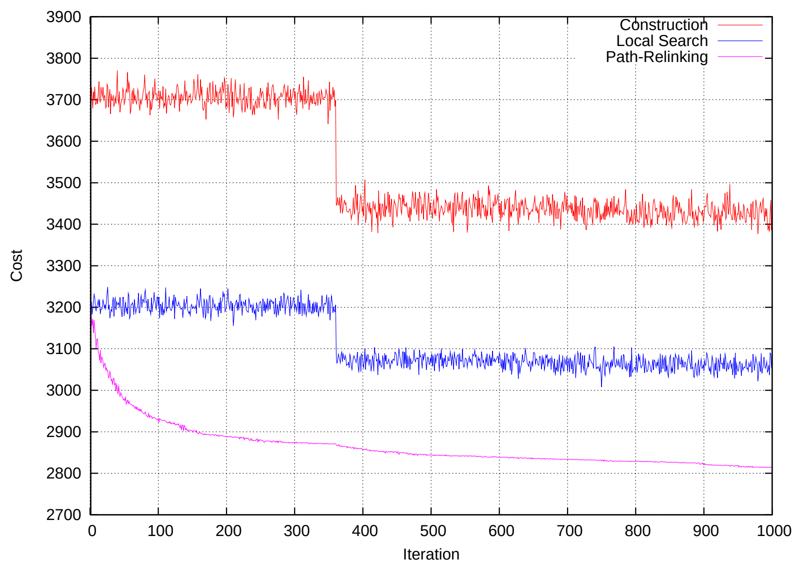

Figure 16: Cost × iteration plot of one execution of MDM-GRASP-PR for instance a400-100 with another random seed

Another experiment was performed to evaluate the time required for GRASP-PR, DM-GRASP-PR and MDM-GRASP-PR to achieve a target solution value. Each strategy was run 100 times (with different random seeds), until a target solution was reached for a specific instance. The instance a400-100 was used as the test case, and two targets were analyzed: an average quality target (value 2834), and a more difficult one (value 2820). Figures 18 and 19 show, for each target, the evaluation of the strategies. For each seed, the time in which the target was reached is plotted. We can observe that in almost all executions, for the two targets, the MDM-GRASP-PR reached the target before the DM-GRASP-PR, which reached the target before the GRASP-PR. For the more difficult target (Figure 19), both DM-GRASP-PR and MDM-GRASP-PR were even more effective, finding it much faster than GRASP-PR.

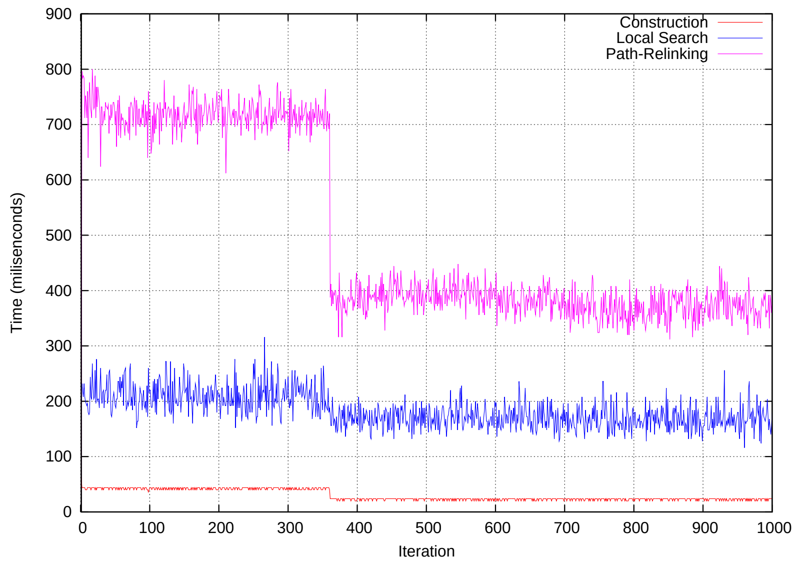

Figure 17: Time $\times$ iteration plot of one execution of MDM-GRASP-PR for instance a400-100 with another random seed

Figures 20 and 21 show another comparison between the three strategies, based on *Time-to-target* (TTT) plots (2), which are used to analyze the behavior of randomized algorithms. These plots basically show the cumulative probability distributions of running times, i.e., $p(\text{computational\_time} < x)$ vs. x.

A TTT plot is generated, initially, by executing an algorithm several times and measuring the time required to reach a solution at least as good as a target solution. In our experiments, each strategy was executed a hundred times. Then, the i-th sorted running time t_i is associated with a probability $p_i = (i - 1/2)/100$ and the points $z_i = (t_i, p_i)$, for $i = 1, ..., 100$ are plotted. Each plotted point indicates the probability (vertical axis) for the strategy to achieve the target solution in the indicated time (horizontal axis). The plots presented in Figures 20 and 21 were generated by the executions of GRASP-PR, DM-GRASP-PR and MDM-GRASP-PR, for instance a400-100, using the same two target solutions used in the previous experiment, respectively: an average value (2834) and a more difficult one (2820).

For the average target, we observe in Figure 20 that GRASP-PR behaves worst than the two other strategies, and that the MDM-GRASP-PR presents better behavior than DM-GRASP-PR. We can see, for example, that the probability for MDM-GRASP-PR to reach the average target in 800s is 100%, for DM-GRASP-PR is approximately 95% and for

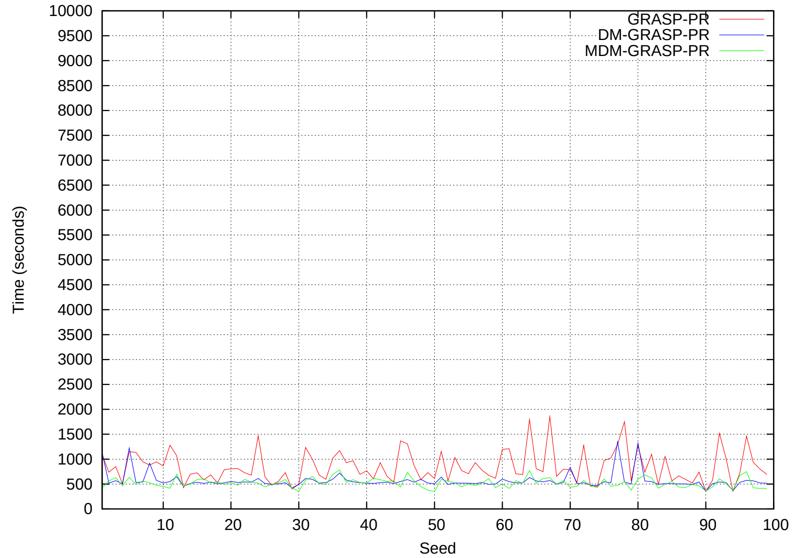

Figure 18: Analysis of convergence to an average target for instance a400-100

GRASP-PR is approximately 58%.

For the difficult target, Figure 21 shows that MDM-GRASP-PR behaves better than DM-GRASP-PR and both presents a better behavior than GRASP-PR. These plots indicate that MDM-GRASP-PR is able to reach difficult solutions faster than DM-GRASP-PR and much faster than GRASP-PR, demonstrating that mining more than once and when the elite set is stable brings robustness to the hybrid strategy.

We can observe that the hybridization of a data mining procedure into a GRASP improved with a path-relinking procedure led the latter to find better quality solutions in less computational time.

We also evaluated the proposed strategies using the tool proposed in (25) to compare two algorithms which are based on stochastic local search. In this work, authors derived a closed form index that gives the probability that one of the algorithms finds a solution at least as good as a given target value in a smaller computation time than the other.

For this experiment, we used again the previous one hundred executions for the instance a400-100. We compared each pair of strategies considering the same average and difficult targets, which have the cost values 2834 and 2820, respectively. We then obtained the probabilities that DM-GRASP-PR finds a solution at least as good as the average and difficult targets in a smaller computation time than GRASP-PR, which are, respectively

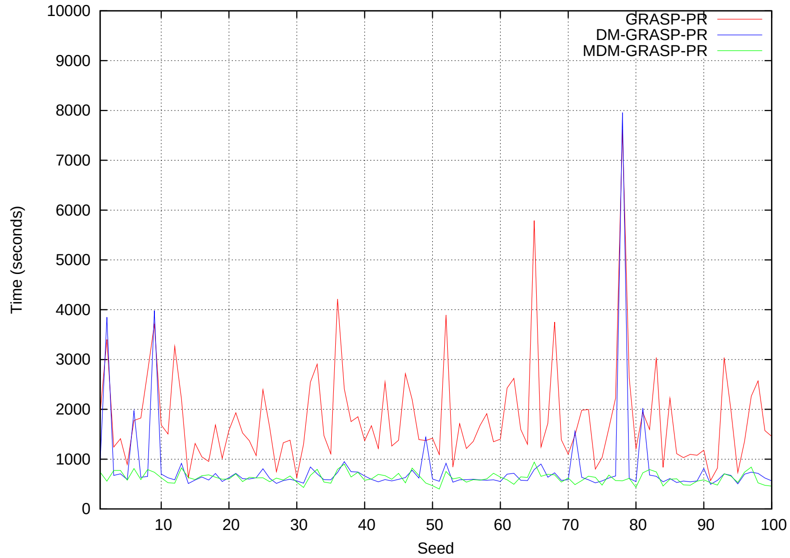

Figure 19: Analysis of convergence to a difficult target for instance a400-100

78.79% and 91.68%. We note that the probability grows with the difficulty of the target. For the difficult target, the data mining strategy presented a much better performance. When comparing MDM-GRASP-PR and GRASP-PR, the obtained probabilities are: 82.37% and 96.98% in favor of the MDM-GRASP-PR strategy. For the comparison between MDM-GRASP-PR and DM-GRASP-PR, the probabilities are 54.24% and 54.76% in favor again of the MDM-GRASP-PR strategy.

In this evaluation, we confirm that incorporating data mining strategies to heuristic methods can improve the performance not only of methods that are memoryless, like the GRASP metaheuristic, but also of methods that incorporate some use of memory, like the path-relinking heuristic. We also observed that the multi data mining approach presented a slightly better behavior when compared to the hybrid strategy which activates the mining process just once.

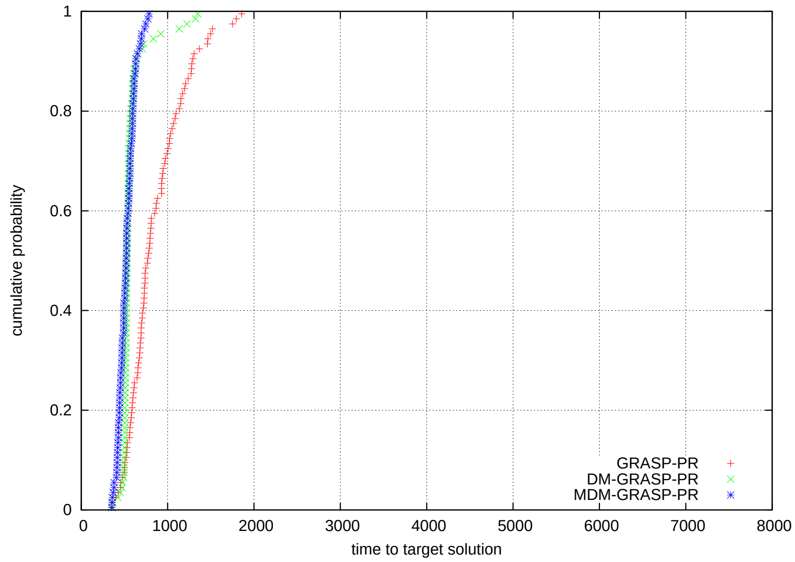

Figure 20: Time-to-target plot to an average target for instance a400-100

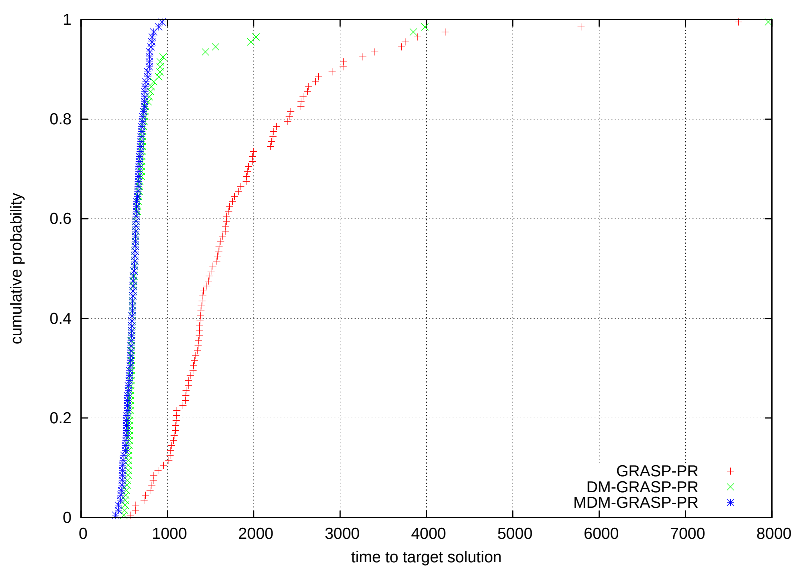

Figure 21: Time-to-target plot to a difficult target for instance a400-100

5 Conclusions and Future Work

Hybrid GRASP metaheuristics which incorporate a data mining procedure has been successfully applied for different combinatorial problems. These hybrid proposals are based on the hypothesis that patterns extracted from sub-optimal obtained solutions using frequent itemset mining could guide the search for better ones.

In this work, we proposed to combine a data mining technique into a GRASP metaheuristic with path-relinking in order to show that not only the traditional GRASP can benefit from using patterns to guide the search, but also GRASP improved with the path-relinking heuristic.

The experimental results showed that the first version of the proposed path-relinking hybrid strategy, called DM-GRASP-PR, was able to obtain better solutions in less computational time than the original GRASP with path-relinking developed to solve the 2-path network design problem, which was a state-of-the-art method for this problem.

In this first version of the path-relinking hybrid GRASP, the data mining process occurred just once. 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 path-relinking hybrid strategy, called MDM-GRASP-PR. This strategy extracts new sets of patterns whenever the elite set changes and becomes stable. The conducted experiments showed that the MDM-GRASP-PR obtained even better results than the DM-GRASP-PR.

These results showed that incorporating a data mining technique is effective, not only to memoryless heuristics, but also to methods that use exchange of information about obtained solutions like the path-relinking strategy.

The results obtained in this work motivate us, as future work, to introduce into others metaheuristics the ideia of extracting patterns from sub-optimal solutions and using them in search procedures. We believe that many others metaheuristics and combinatorial optimization problems can benefit from the incorporation of data mining techniques. The good results obtained in this work motivate us, as future work, to introduce data mining procedures into other metaheuristics. We believe that many other metaheuristics, such as tabu search and genetic algorithms, can benefit from the incorporation of data mining techniques, reaching good solutions for many other combinatorial optimization problems in reasonable computational times.

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 R. Aiex, M. G. C. Resende, and C. C. Ribeiro, *TTT plots: a perl program to create time-to-target plots*, Optimization Letters, 4 (2007), pp. 355–366.

3 G. Dahl and B. Johannessen, *The 2-path network problem*,

Networks, 43 (2004), pp. 190–199.

4 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.

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

6 P. Festa and M. G. C. Resende, *An annotated bibliography of GRASP Part I: Algorithms*, International Transactions in Operational Research, 16 (2009), pp. 1–24.

7 P. Festa and M. G. C. Resende, *An annotated bibliography of GRASP Part II: Applications*, International Transactions in Operational Research, 16 (2009), pp. 131–172.

8 C. Fleurent and F. Glover, *Improved Constructive Multistart Strategies for the Quadratic Assignment Problem Using Adaptive Memory*, INFORMS Journal on Computing, 2 (1999), pp. 198-204.

9 F. Glover, M. Laguna, and R. Martí, *Fundamentals of scatter search and path-relinking*, Control and Cybernetics 19, pp. 653–684, 1977.

10 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.

11 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.

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

13 M. Laguna and R. Martí, *GRASP and path relinking for 2-layer straight line crossing minimization*, INFORMS Journal on Computing **11** (1999), pp. 44–52.

14 S. Lin and B.W. Kernighan, *An effective heuristic algorithm for the traveling salesman problem*, Operations Research, 21 (1973), pp. 498–516.

15 A. Lodi, K. Allemand, and T. M. Liebling, *An evolutionary heuristic for quadratic 0-1 programming*, European Journal of Operational Research, 119 (1999), pp. 662–670.

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

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

18 A. Plastino, E. R. Fonseca, R. Fuchshuber, S. L. Martins, A. A. Freitas, M. Luis, and S. Salhi, *A hybrid data mining metaheuristic for the p-median problem*, Proceedings of the SIAM International Conference on Data Mining, pp. 305-316, 2009.

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

20 M. G. C. Resende and C. C. Ribeiro, *GRASP with path-relinking: Recent advances and applications*, Metaheuristics: Progress as Real Problem Solvers (T. Ibaraki et al. editors), (2005), 29–63.

21 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.

22 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.

23 C. C. Ribeiro, S. L. Martins, and I. Rosseti, *Metaheuristics for optimization problems in computer communications*, Computer Communications, 30 (2007), pp. 656-669.

24 C. C. Ribeiro and I. Rosseti, *Efficient parallel cooperative implementations of GRASP heuristics*, Parallel Computing 33 (2007), pp. 21-35.

25 C. C. Ribeiro, I. Rosseti, and R. Vallejos, *On the use of run time distributions to evaluate and compare stochastic local search algorithms*, Proceedings of the Engineering Stochastic Local Search Algorithms Workshop, Lecture Notes in Computer Science 5752, pp. 16–30, 2009.

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

27 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, Lecture Notes in Computer Science 3636, pp. 116–127, 2005.

28 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.

29 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.

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

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