Incorporando Técnicas de Mineração de Dados à Metaheurística GRASP

Marcos Henrique Fonseca Ribeiro

Dissertação de Mestrado submetida ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do título de Mestre em Computação.

Orientador: Alexandre Plastino de Carvalho
Co-orientadora: Simone de Lima Martins

Niterói, Novembro de 2005.

Incorporando Técnicas de Mineração de Dados à Metaheurística GRASP

Marcos Henrique Fonseca Ribeiro

Dissertação de Mestrado submetida ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do título de Mestre em Computação.

Aprovada por:

Alexandre Plastino de Carvalho / IC-UFF (Presidente)

Simone de Lima Martins / IC-UFF

Celso da Cruz Carneiro Ribeiro / IC-UFF

Luis Satoru Ochi / IC-UFF

Cid Carvalho de Souza / UNICAMP

Niterói, Novembro de 2005.

Aos meus pais e irmãs.

Resumo da Tese apresentada à UFF como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação (M.Sc.)

Incorporando Técnicas de Mineração de Dados à Metaheurística GRASP

Marcos Henrique Fonseca Ribeiro

Novembro/2005

Orientadores: Alexandre Plastino de Carvalho e Simone de Lima Martins
Programa de Pós-Graduação em Ciência da Computação

Este trabalho investiga a eficiência da incorporação de técnicas de Mineração de Dados à metaheurística GRASP no intuito de introduzir memória à mesma, gerando assim uma versão híbrida da metaheurística GRASP, denominada GRASP-MD.

Para a validação da proposta, foi utilizado o Problema do Empacotamento de Conjuntos (PEC) e diferentes versões da metaheurística híbrida foram testadas e analisadas.

Experimentos computacionais realizados com o objetivos de comparar a metaheurística GRASP tradicional com as difreentes versões híbridas mostraram que a utilização de padrões minerados a partir de um conjunto elite de soluções gerou melhores resultados. Além disso, foi possível perceber que as estratégias híbridas foram capazes de alcançar boas soluções em um menor tempo de processamento.

Abstract of Thesis presented to UFF as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

Incorporating Data Mining Techniques into the GRASP Metaheuristic

Marcos Henrique Fonseca Ribeiro

November/2005

Advisors: Alexandre Plastino de Carvalho and Simone de Lima Martins
Department: Computer Science

In this work, we investigate the efficiency of incorporating Data Mining techniques to the GRASP metaheuristic in order to introduce memory to this strategy, generating a hybrid version of GRASP, called GRASP-MD.

The Set Packing Problem (SPP) was used to validate this proposal and different versions of the hybrid metaheuristic were tested and analyzed.

Computational experiments, comparing traditional GRASP and different hybrid approaches, showed that employing patterns mined from an elite set of solutions conducted to better results. Besides, additional performance experiments evidenced that data mining strategies accelerate the process of finding good solutions.

Palavras-chave

1. Metaheurística Híbrida
2. GRASP
3. Mineração de Dados
4. Problema do Empacotamento de Conjuntos
5. Otimização Combinatória

Glossário

DCI	:	*Direct Count & Intersect*;
DCP	:	*Direct Count & Prune transactions*;
MP	:	Estratégia do Maior Padrão;
MPS	:	Estratégia do Maior Padrão para cada Suporte;
nMM	:	Estratégia dos n Maiores Padrões Maximais;
nMP	:	Estratégia dos n Maiores Padrões;
FH	:	Fase Híbrida do GRASP-MD;
GCP	:	Geração do Conjunto de Padrões;
GRASP	:	*Greedy Randomized Adaptative Search Procedures*;
GRASP-MD	:	Metaheurística GRASP Híbrida com Mineração de Dados;
LRC	:	Lista Restrita de Candidatos (*Restricted Candidate List*);
MCF	:	Mineração de Conjuntos Freqüentes (*Frequent Itemset Mining*);
MD	:	Mineração de Dados (*Data Mining*);
MRA	:	Mineração de Regras de Associação (*Association Rule Mining*);
PEC	:	Problema do Empacotamento de Conjuntos (*Set Packing Problem*);
RA	:	Regras de Associação;
VND	:	*Variable Neighborhood Descent*

Sumário

Resumo **iii**

Abstract **iv**

Glossário **vi**

1 Introdução **1**

2 O Problema do Empacotamento de Conjuntos - PEC **4**

2.1 Formulação 4

2.2 Exemplo do PEC 5

2.3 Trabalhos relacionados ao PEC 6

3 GRASP para o Problema de Empacotamento de Conjuntos **8**

3.1 GRASP 8

3.2 GRASP para o PEC 10

3.2.1 Fase de Construção 10

3.2.2 Fase de Busca Local 12

4 Mineração de Dados **16**

4.1 Mineração de Regras de Associação 16

4.2 Mineração de Conjuntos Freqüentes 18

4.2.1 O Algoritmo *DCI* 20

4.2.2 O Algoritmo *FPmax** 21

5 A Estratégia Híbrida GRASP-MD 23

5.1 A Estratégia do Maior Padrão (MP) 25

5.2 A Estratégia do Maior Padrão para cada Suporte (MPS) 26

5.3 A Estratégia dos n Maiores Padrões (nMP) 28

5.4 A Estratégia dos n Maiores Padrões Maximais (nMM) 29

6 Resultados Experimentais 32

6.1 Instâncias Utilizadas 32

6.2 Ambiente de Execução 34

6.3 Comparação Entre as Estratégias 35

7 Conclusões Finais e Trabalhos Futuros 55

Referências Bibliográficas 58

Lista de Figuras

2.1 Matriz T de restrições 5

4.1 Base de Dados Transacional 18

6.1 Análise de freqüência das estratégias híbridas. 43

6.2 Análise de freqüência dos algoritmos nMM e GRASP. 44

6.3 Alvo fácil para a instância 9. 46

6.4 Alvo difícil para a instância 9. 47

Lista de Tabelas

5.1 Características dos padrões de cada estratégia 31

6.1 Características das Instâncias 33

6.2 Comparação entre as estratégias GRASP e híbrida MP 36

6.3 Comparação entre as estratégias híbridas MP e MPS 38

6.4 Comparação entre as estratégias híbridas MPS e nMP 39

6.5 Comparação entre as estratégias híbridas nMP e nMM 41

6.6 Percentual médio em relação ao melhor valor conhecido 42

6.7 Diferença percentual entre o melhor valor obtido por cada estratégia e o melhor valor conhecido 45

6.8 Avaliação do tamanho do conjunto elite 48

6.9 Tamanhos médios dos padrões utilizados 49

6.10 Número médio de padrões utilizados 51

6.11 Suporte médio dos padrões utilizados 52

6.12 Número médio percentual de interseções entre os padrões utilizados . 54

Lista de Algoritmos

1 *GRASP* () 10

2 *AlgoritmoConstrução* (α) 12

3 *AlgoritmoBuscaLocal* (*Solução*) 15

4 *AlgoritmoConstruçãoAdaptada* (p) 25

Capítulo 1

Introdução

Mineração de Dados (MD) é um tema da Ciência da Computação cujos processos têm sido utilizados em uma ampla variedade de áreas do conhecimento, tais como *marketing*, finanças, saúde, educação e segurança. Refere-se à extração de conhecimento potencialmente útil de bases de dados na forma de regras e padrões, dentre os quais se destacam as regras de associação, os padrões de seqüência, as regras de classificação e o agrupamento de dados [5, 17]. Os principais objetivos do uso de MD são a descrição e a predição. Na descrição, os padrões e regras descrevem características importantes dos dados com os quais se está trabalhando, enquanto na predição deseja-se prever o valor desconhecido de um determinado atributo, baseado na análise histórica dos dados armazenados na base de dados.

Resumidamente, o objetivo deste trabalho é investigar a utilização de padrões, extraídos através de técnicas de MD, que representam características de soluções sub-ótimas de um problema de otimização combinatória, para guiar a obtenção de novas e melhores soluções na resolução de tal problema. Em suma, a idéia é, através da introdução de técnicas de Mineração de Dados, incluir aprendizado de máquina em uma técnica de otimização.

A forma escolhida para guiar a obtenção de novas soluções, a partir dos padrões obtidos com a MD, foi a fixação de variáveis. Alguns trabalhos que utilizam

a fixação de variáveis para reduzir o espaço de busca em problemas de otimização combinatória e que conduzem a soluções finais de boa qualidade podem ser encontrados na literatura. Pardalos e Rodgers [27] desenvolveram um algoritmo do tipo *Branch-and-Bound* para solucionar problemas quadráticos binários que incorpora um método para fixar variáveis durante o seu processo de execução. Lodi *et al.* [19] apresentaram uma heurística para solucionar o mesmo problema baseada em Algoritmos Genéticos e *Scatter Search*. Também neste trabalho, incorporaram fixação de variáveis comparando membros das populações obtidas nas iterações desta heurística. Em [11], Glover estuda a fixação de variáveis aplicada às técnicas *Scatter Search* e *Path-Relinking*. Os bons resultados da literatura motivaram a utilização da abordagem de fixação de variáveis na técnica de otimização desenvolvida neste trabalho.

A técnica de otimização escolhida para tal estudo foi a metaheurística *Greedy Randomized Adaptive Search Procedure* (GRASP) [7], que tem como característica importante, na sua forma clássica, a ausência de memória e aprendizado a respeito das etapas anteriores – a menos da melhor solução para o problema encontrada até então – quando se considera uma etapa qualquer de seu processo de execução. A idéia central da proposta apresentada nesta dissertação consiste em criar uma versão híbrida da metaheurística GRASP, aqui denominada **GRASP-MD**, que incorpora técnicas de MD. Esta combinação de técnicas ocorre da seguinte forma. Inicialmente, são armazenadas soluções sub-ótimas encontradas em um número significativo de iterações da metaheurística GRASP. Em seguida, dispara-se um processo de MD que extrai padrões que ocorrem com freqüência dentro de um subconjunto destas soluções, composto pelas soluções que apresentam melhor qualidade – chamado conjunto elite. Então, de acordo com alguma estratégia de utilização, aproveitam-se estes padrões na construção de soluções nas iterações subseqüentes.

O processo de obtenção dos padrões utilizado aqui é conhecido como Mineração de Conjuntos Freqüentes (MCF) [13], um subproblema daquele conhecido como Mineração de Regras de Associação (MRA) [5, 17]. Desde 1993, muitos algoritmos eficientes têm sido propostos para a tarefa de MCF [1, 2, 13].

De acordo com a taxonomia proposta em [34] para metaheurísticas híbridas, a proposta GRASP-MD pode ser classificada como uma metaheurística encadeada (*relay*) e de alto nível (*high level*). Pode ser considerada de alto nível porque as técnicas de MD e GRASP são auto contidas, não havendo mistura de códigos e pode ser considerada encadeada porque as técnicas GRASP, MD e GRASP novamente são aplicadas de maneira linear, onde a saída de uma é aplicada à entrada da outra, em um esquema semelhante a uma linha de produção.

Os bons resultados obtidos em [6], aplicando-se uma versão da metaheurística GRASP para o problema conhecido como Problema do Empacotamento de Conjuntos (PEC), ou *Set Packing Problem*, motivaram a utilização deste problema para validar a proposta GRASP-MD. Foi possível com esta escolha, avaliar de forma mais precisa a parcela de contribuição da hibridação para a melhoria dos resultados obtidos com relação à técnica não híbrida, apresentada em [6].

O restante desta dissertação está organizado da seguinte forma. No Capítulo 2, o problema PEC é introduzido e formulado, e são apontados outros trabalhos que o abordam. Uma implementação da metaheurística GRASP para o PEC, baseada em uma das versões propostas em [6] é apresentada no Capítulo 3. Uma descrição das técnicas de Mineração de Conjuntos Freqüentes utilizadas para a hibridação são encontrados no Capítulo 4. No Capítulo 5, diferentes abordagens para a hibridação são propostas e suas respectivas implementações em alto nível são apresentadas e discutidas. Os testes computacionais e resultados experimentais são reportados no Capítulo 6. Por fim, no Capítulo 7, são feitas as conclusões finais e sugestões de trabalhos futuros.

Capítulo 2

O Problema do Empacotamento de Conjuntos - PEC

2.1 Formulação

O Problema do Empacotamento de Conjuntos (*Set Packing Problem*) [6] é um problema clássico de otimização combinatória, pertencente à classe NP-difícil [10] e similar ao problema conhecido como Problema de Cobertura de Conjuntos (*Set Covering Problem*) [22]. Uma definição do PEC é dada a seguir. Seja $I = \{1, \ldots, m\}$ um conjunto finito e seja $\{I_j\}$ para $j \in J = \{1, \ldots, n\}$ uma coleção de subconjuntos de I. Diz-se que $P \subseteq J$ empacota I se $I_j \cap I_k = \emptyset \; \forall j, k, j \neq k$. Este problema pode ser formulado como um problema binário, considerando-se uma matriz de incidência tal que

$$t_{ij} = \begin{cases} 1, se\ i \in I_j \\ 0, caso\ contrário \end{cases} \quad \text{e } x_j = \begin{cases} 1, se\ j \in P \\ 0, caso\ contrário \end{cases}.$$

Matematicamente, o PEC pode ser formulado como a seguir, onde o parâmetro c_j representa o peso ou o custo do item j, isto é, representa o ganho de se colocar o item j no empacotamento.

$$\max z = \sum_{j \in I} c_j x_j, \tag{2.1}$$

$$x_j \in \{0, 1\}, \ \forall j, \tag{2.2}$$

$$\sum_{j=1}^{n} t_{ij} x_j \leq 1, \ i = 1, \ldots, m \tag{2.3}$$

Em outras palavras, deve-se encontrar um empacotamento P que maximize a soma dos custos de seus itens, sem violar as restrições definidas na matriz de incidência. Cada restrição define itens que não podem estar simultaneamente no empacotamento P.

2.2 Exemplo do PEC

Apresenta-se a seguir um exemplo do PEC. Sejam $I = \{1, 2, 3, 4, 5, 6\}$ o conjunto de itens disponíveis e $c = \{2, 3, 4, 1, 7, 1\}$ o conjunto de seus respectivos custos. A matriz $T_{6\times4}$, apresentada na Figura 2.1, representa o conjunto de restrições aplicadas ao problema. Como a matriz T possui 4 colunas, o exemplo contém esse mesmo número de restrições. Em uma determinada coluna, os valores das linhas (de 1 a 6) representam a presença (1) ou não (0) dos itens de 1 a 6 em uma determinada restrição.

$$T = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

Figura 2.1: Matriz T de restrições

Assim, ao analisar a primeira linha de T, nota-se que o elemento 1 está presente somente nas restrições 1 e 2. Da mesma forma, analisando-se a quarta coluna da matriz, sabe-se que os elementos 3, correspondente à terceira linha, e 5, correspondente

à quinta linha, não podem ocorrer simultaneamente em uma determinada solução para esta instância do PEC.

O vetor binário $x = [1, 0, 0, 1, 1, 0]$, que representa o empacotamento $P = \{1, 4, 5\}$ é uma solução viável para o problema, pois não viola nenhuma restrição de T, ou seja, os itens que o compõem não têm suas respectivas posições definidas com o valor 1 simultaneamente em nenhuma coluna na matriz.

A análise das restrições representadas por T conduz à observação de que o elemento 4, por não estar presente em nenhuma restrição, uma vez que todos os valores correspondentes à sua linha são iguais a zero, pode participar de qualquer solução do problema, sem inviabilizá-la. O que também significa que certamente estará presente na solução ótima.

O indicador de qualidade da solução é dado pela soma dos custos correspondentes aos itens presentes na solução, representado por z, valor que se deseja maximizar. Assim, quanto maior este valor, melhor a solução encontrada. Para a solução do exemplo acima, $z = 2 + 1 + 7 = 10$.

Uma outra solução possível, e de melhor qualidade, para a mesma instância seria $P = \{2, 4, 5\}$, com $z = 11$, enquanto o empacotamento $P = \{1, 4, 5, 6\}$ não seria uma solução viável para o problema, por violar a primeira restrição, que impede a presença simultânea de 1 e 6 em um empacotamento.

Vale ainda observar que uma característica importante do PEC é que a ordem com que os itens são representados na solução não é relevante. Desta forma, os conjuntos $P_1 = \{1, 4, 5\}$ e $P_2 = \{5, 1, 4\}$, com somas dos custos $z_1 = z_2 = 10$, representam exatamente a mesma solução para o exemplo anterior.

2.3 Trabalhos relacionados ao PEC

Quando o interesse é apenas maximizar o número de itens de um empacotamento em um PEC, sem preferência por nenhum item específico, pode-se utilizar a mesma formulação definida anteriormente, porém definindo-se o custo relativo a cada item

como possuindo o valor 1. Delorme *et al.* [6] utilizaram esta abordagem para algumas instâncias reais de um problema ferroviário. O PEC pode ainda ser particularizado como um problema conhecido como Problema de Empacotamento de Nós (*Node Packing Problem*) [22], onde as restrições consideram apenas pares de itens. A partir de uma instância do PEC, podem ser geradas todas as C_n^2 combinações dos n itens pertencentes a uma restrição do problema. Cada uma destas combinações é considerada, por sua vez, como uma restrição no problema reformulado como um Problema de Empacotamento de Nós. Entretanto, esta abordagem conduz a um aumento significativo na quantidade de restrições com relação ao problema original, devido ao grande número de combinações que podem ser geradas a partir de uma dada restrição na formulação original do problema.

Outros trabalhos que abordam o mesmo problema são citados a seguir. Padberg [26], que desenvolveu um algoritmo *branch-and-cut* capaz de resolver de maneira exata instâncias de pequeno tamanho deste problema. Guo *et al.* [16] formularam o Problema Combinatório de Requisições em Leilão (*Combinatorial Auction Brokering Problem*) como um PEC e aplicaram *Simulated Annealing* na sua resolução. Mingozzi *et al.* [20] calcularam limites para o Problema de Escalonamento de Projetos com Restrições de Recursos (*Resource Constrained Project Scheduling Problem*) com um método guloso, formulando-o como um PEC. Zwaneveld *et al.* [35] formularam um problema de rotas em estradas de ferro como um PEC e o resolveram utilizando o método *branch-and-cut*. Delorme *et al.* [6] desenvolveram algumas variantes da metaheurística GRASP para este problema e efetuaram diversos testes computacionais usando instâncias geradas aleatoriamente e também instâncias de problemas reais. Existe ainda uma abordagem utilizando um algoritmo Colônia de Formigas para o PEC desenvolvido e testado por Gandibleux *et al.* [9].

No próximo capítulo, é descrita uma implementação da metaheurística GRASP para a resolução do PEC, baseada em um dos procedimentos GRASP desenvolvidos e apresentados por Delorme *et al.* em [6]. Esta implementação é utilizada como base para o desenvolvimento da proposta desta dissertação, uma versão híbrida da técnica GRASP com técnicas de Mineração de Dados, descrita posteriormente.

Capítulo 3

GRASP para o Problema de Empacotamento de Conjuntos

O conjunto de técnicas heurísticas genéricas conhecidas como metaheurísticas desempenham papel importante na busca por soluções de qualidade em problemas da classe NP-difícil. O emprego de metaheurísticas a este tipo de problema possibilita, através do emprego de estratégias de busca eficientes, obter boas soluções em tempo viável. Uma destas metaheurísticas é aquela conhecida como *Greedy Randomized Adaptative Search Procedure* (GRASP), proposta por Feo e Resende em [7].

3.1 GRASP

GRASP é um processo iterativo, onde cada iteração é composta basicamente de duas fases. A primeira é chamada etapa de construção, por sua vez também um processo iterativo, onde é obtida uma solução viável para o problema utilizando uma heurística gulosa e aleatória. Os itens candidatos a compor uma solução são ordenados em uma lista segundo uma função de avaliação que determina, a partir de critérios definidos na elaboração da metaheurística, o seu potencial de contribuição para a construção de uma solução aproximada de boa qualidade. Após feita a

avaliação e ordenação dos candidatos, esta lista é reduzida para conter apenas os itens de melhor avaliação, isto é, os itens com maior potencial para gerar uma solução de boa qualidade e, dentre estes itens, um item é selecionado aleatoriamente para ser incluído na solução. Após a seleção de um item uma nova lista de candidatos é criada com os processos de ordenação, restrição e escolha aleatória feitos para os candidatos restantes. Este procedimento é repetido até que não haja mais possibilidades de seleção de nenhum item, isto é, que a lista não contenha nenhum item. Condições que garantam a viabilidade da solução gerada por este processo têm que ser verificadas e, se necessário, as medidas cabíveis para que estas condições sejam atendidas têm que ser tomadas como, por exemplo, descartar da lista de candidatos itens que inviabilizem a solução, se incluídos na mesma.

A solução gerada pela etapa de construção não constitui garantidamente um ótimo local. A segunda etapa, chamada busca local, percorre, a partir da solução encontrada na primeira etapa, uma vizinhança composta por soluções obtidas a partir de variações sobre a mesma, em busca de soluções melhores que a construída originalmente. A solução obtida ao final de uma iteração GRASP é confrontada com a melhor solução já encontrada dentre todas as iterações até então e caso a solução atual a supere em qualidade, passa a ser considerada a melhor solução. Ambas etapas, de construção e busca local, são repetidas iterativamente até que um critério de parada seja ativado e a execução da técnica GRASP seja concluída, trazendo como resultado a melhor solução encontrada entre todas iterações efetuadas.

Em suma, a idéia básica do GRASP é, a partir de uma solução inicial, aprimorá-la sucessivamente. Este processo é aplicado repetidas vezes para que se possa ampliar as chances de se encontrar a solução ótima para determinado problema de otimização. Uma característica da metaheurística GRASP, importante para este estudo, é que cada iteração é independente das que foram executadas anteriormente, isto é, a cada iteração GRASP a etapa de construção gera uma nova solução inicial completamente independente da solução resultante na iteração anterior, o que caracteriza um desprovimento de memória ou de aprendizado na técnica. Esta característica representa um ponto para aperfeiçoamento, explorado pela técnica híbrida proposta nesta dissertação, descrita no Capítulo 5.

Um esquema da versão clássica da metaheurística GRASP pode ser visto no pseudo-código do Algoritmo 1. Na linha 1, a variável que é utilizada para armazenar a solução final do GRASP é inicializada. Nas linhas de 2 a 6, as duas fases que compõem a metaheurística são executadas iterativamente até que um critério de parada seja atingido. Este critério usualmente consiste em um número predefinido de iterações ou um limite máximo de tempo de CPU. Na linha 3, um algoritmo de construção é executado, gerando uma solução inicial para posterior refinamento, executado na linha 4. Na linha 5, solução refinada é então confrontada com a melhor solução obtida até então pelo GRASP e a melhor entre estas é armazenada. Por fim, na linha 7, a melhor solução entre todas obtidas pelas iterações GRASP é retornada.

Algoritmo 1 *GRASP* ()

```
1: MelhorSolução ← φ;
2: repita
3:    SoluçãoInicial ← AlgoritmoConstrução;
4:    SoluçãoMelhorada ← AlgoritmoBuscaLocal(SoluçãoInicial);
5:    MelhorSolução ← melhor(MelhorSolução, SoluçãoMelhorada);
6: até critério de parada satisfeito;
7: retorna MelhorSolução;
```

A seguir, é apresentada a implementação de uma versão desta metaheurística para o PEC, utilizada no estudo desta dissertação e inspirada no trabalho desenvolvido por Delorme *et al* [6].

3.2 GRASP para o PEC

3.2.1 Fase de Construção

O procedimento de construção implementado para esta etapa é inspirado naquele chamado *greedy2* apresentado por Delorme *et al.* em [6]. Algumas definições são necessárias para o melhor entendimento dos procedimentos descritos nesta seção. Seja $X = [x_1, \ldots, x_n]$ o vetor binário que representa uma solução para o PEC, onde $x_i = 1$, se o item $i \in I$ pertence à solução e $x_i = 0$, se o item i não está presente na solução. Seja $I_{it} \subseteq I$, o conjunto dos itens não presentes na solução X

do problema e que, em nenhuma das restrições da instância do problema, ocorrem simultaneamente com os itens já presentes em X. Em outras palavras, I_{it} é o subconjunto de I composto pelos itens ainda não pertencentes a X e que podem ser incluídos na solução sem inviabilizá-la. O conjunto I_{it} será chamado deste ponto em diante de conjunto de candidatos da etapa construtiva.

O processo construtivo parte da solução factível inicial $x_i = 0, \forall i \in I$. Em cada iteração da fase de construção, os elementos i pertencentes ao conjunto de candidatos I_{it} são avaliados priorizando-se os itens de maior custo, envolvidos no menor número possível de restrições. Para tanto, aplica-se a função $e(i) = c_i / \sum_{j \in J} t_{i,j}, \forall i \in I_{it}$, que computa a razão entre o custo associado ao item i e o número de restrições nas quais o mesmo está presente. Estes itens são então organizados em uma lista L, em ordem decrescente de seus valores de $e(i)$.

De L, é criada uma outra lista, chamada Lista Restrita de Candidatos (LRC), composta apenas pelos l primeiros itens de L, onde

$$l = \begin{cases} \lceil \alpha \times n_{it} \rceil, se\ \alpha > 0 \\ 1, se\ \alpha = 0, \end{cases} \tag{3.1}$$

n_{it} é o número de itens de L e $\alpha \in [0, 1]$ é um parâmetro de entrada do algoritmo GRASP. A partir daí, um elemento é escolhido aleatoriamente dentro de LRC para ser incluído na solução parcial do problema. Se $\alpha = 0$, a LRC é composta por apenas 1 elemento, então o critério de escolha do algoritmo é puramente guloso. Se $\alpha = 1$, todos os elementos de L participarão da escolha, tornando a mesma totalmente aleatória.

Após um item i ser incluído no vetor solução em construção, todos os elementos de I_{it} presentes em restrições que contêm i são excluídos do conjunto de candidatos, gerando um novo conjunto I_{it}, de forma a garantir que a solução a ser construída seja factível. Neste ponto, encerra-se a iteração corrente e o processo é repetido até que $I_{it} = \phi$.

O Algoritmo 2 traz o pseudo-código do procedimento construtivo implementado. Na linha 1, o conjunto de candidatos é iniciado com todos os itens, enquanto na linha 2, a solução inicial é definida considerando-se todos os elementos ausentes na

solução. A função de avaliação $e(i)$ é calculada para todos os candidatos na linha 3. Na linha 4, o conjunto I_{it} é ordenado, de forma decrescente, de acordo com as avaliações $e(i)$ de seus itens. As linhas de 5 a 16 compõem as iterações construtivas, que são executadas até que não haja mais candidatos disponíveis. O cálculo do limite l é efetuado nas linhas de 6 a 10 enquanto a LRC é construída na linha 11. As linhas 12 e 13 trazem, respectivamente, a escolha aleatória do item i^* de LRC e sua inserção na solução. A remoção do item i^* de I_{it} é feita na linha 14, enquanto na linha 15 todos os itens de I_{it} afetados pelas restrições que contêm i^* são também removidos.

Algoritmo 2 *AlgoritmoConstrução* (α)

1: $I_{it} \leftarrow I$;
2: $x_i \leftarrow 0, \forall i \in I_{it}$;
3: $e(i) \leftarrow c_i / \sum_{j \in J} t_{i,j}, \forall i \in I_{it}$;
4: $I_{it} \leftarrow Ordena(I_{it}, e(i))$
5: **enquanto** $(I_{it} \neq 0)$ **faça**
6: **se** $\alpha > 0$ **então**
7: $l \leftarrow \lceil \alpha \times n_{it} \rceil$;
8: **senão**
9: $l \leftarrow 1$;
10: **fim se**
11: $LRC \leftarrow l$ primeiros elementos de I_{it};
12: $i^* \leftarrow escolhaAleatória(LRC)$;
13: $x_{i^*} \leftarrow 1$;
14: $I_{it} \leftarrow I_{it} \backslash \{i^*\}$;
15: $I_{it} \leftarrow I_{it} \backslash \{i : \exists j \in J, t_{i,j} + t_{i^*,j} > 1\}$;
16: **fim enquanto**
17: **retorna** X.

3.2.2 Fase de Busca Local

Da forma como uma solução é construída no GRASP, não existem garantias de que esta represente um ótimo local. Portanto, uma busca pela vizinhança desta solução deve ser efetuada na tentativa de melhorá-la. Na implementação aqui apresentada, define-se como vizinha de uma solução aquela que se pode obter a partir desta removendo-se k itens presentes e inserindo-se p que não estão presentes na mesma, em um processo chamado *trocas k-p* (*k-p exchanges*). Observa-se aqui que se deve garantir que as trocas efetuadas entre os elementos da solução gerem

uma nova solução viável, isto é, que não viole nenhuma restrição da instância do problema. Devido ao alto custo computacional exigido por este tipo de busca, a vizinhança k-p de uma solução, composta por soluções obtidas por trocas k-p, não é necessariamente toda percorrida, como normalmente é feito nas implementações da metaheurística GRASP.

Assim, algumas adaptações foram feitas no modelo original para atender às necessidades encontradas na resolução do PEC. Na proposta deste trabalho, a etapa de busca local foi substituída por uma técnica inspirada naquela conhecida na literatura como *Variable Neighborhood Descent* (VND) [21]. Na VND, em vez de uma, percorrem-se diferentes vizinhanças. A justificativa para incorporação deste procedimento é que, uma vez que a busca local não necessariamente percorre uma vizinhança até que a mesma esteja exaurida, ao se permitir buscar em vizinhanças diferentes, uma estratégia para ampliar as chances de se encontrar um vizinho melhor esteja sendo adotada, através da variabilidade das trocas efetuadas.

Foram adotadas, na implementação deste trabalho, as vizinhanças 1-2 e 2-1, cuja ordem de utilização é determinada aleatoriamente para cada iteração GRASP. Também de forma aleatória são escolhidos quais k itens são retirados e quais p itens são incluídos na solução. A garantia de viabilidade da solução gerada pelas trocas k-p é feita por procedimentos que controlam a lista de candidatos a participar da solução, efetuando inclusões e podas na mesma. Sempre que os k itens são retirados da solução, elementos podem ser incluídos na lista de candidatos. Quando os p itens são inseridos na solução, a lista de candidatos deve ser podada, em operações similares àquelas apresentadas nas linhas 10 e 11 do algoritmo de construção, apresentado na Figura 2. Após determinada a ordem das vizinhanças, a busca local é iniciada na primeira delas. Se for encontrada uma solução melhor que a solução corrente na busca em uma determinada vizinhança, esta solução vizinha é adotada como solução base para uma nova busca e o processo de busca é reiniciado para esta nova solução. Se a vizinhança for exaurida sem que se obtenha melhoria na solução, uma nova busca é feita na segunda vizinhança. Se na segunda vizinhança for conseguida uma melhoria na solução, todo o processo é reiniciado, retornando à primeira vizinhança, tendo como base esta nova solução. Vale ressaltar que é adotada uma estratégia de

se reiniciar o processo sempre na primeira melhoria encontrada, interrompendo a busca corrente.

A busca local é finalizada quando nenhuma melhoria é alcançada após serem percorridas ambas as vizinhanças em sua totalidade. Ao final da etapa de busca local a solução resultante é confrontada com a melhor solução encontrada até então. Se possuir melhor qualidade, passa a ser a melhor solução corrente.

O Algoritmo 3 traz o pseudo-código do procedimento de busca local. Na linha 1, a ordem de busca nas vizinhanças a serem percorridas é aleatoriamente escolhida, enquanto na linha 2, a primeira delas é definida. Uma variável de controle para identificar o estado de melhoria ou não é iniciada com o valor 1 (verdadeiro) na linha 3. As linhas de 4 a 16 compõem as iterações de busca local, que são executadas até que nenhum dos vizinhos de uma determinada solução melhore seu valor. Na linha 5, a busca local é realizada na primeira vizinhança e, se um melhor vizinho for encontrado, a solução corrente é substituída por este, caso contrário, a vizinhança é trocada na linha 9 e outra busca é efetuada na linha 10. Se um vizinho de melhor qualidade é encontrado, a solução é atualizada com o valor deste na linha 12 e a vizinhança é trocada novamente na linha 13. Se não houver tal vizinho, a busca local é finalizada.

Algumas observações merecem ser feitas a respeito do funcionamento e comportamento desta técnica. Destaca-se o fato de não haver garantias de que a solução final obtida ao término de sua execução seja o ótimo global. Faz-se necessário então um estudo estatístico da eficiência da metaheurística utilizando-se um conjunto de instâncias de testes e tendo como base comparativa de desempenho um outro método de resolução, de preferência um que forneça o valor exato da solução ótima global.

Outra característica importante é a independência entre o resultado construído em uma dada iteração e aqueles obtidos nas iterações anteriores. Essa ausência de memória entre iterações impossibilita que a técnica utilize o histórico de seu desempenho para extrair informações potencialmente importantes a respeito das características das soluções encontradas no decorrer de sua execução, na tentativa

de se obter resultados mais eficientes. Conforme dito anteriormente, a proposta deste trabalho é desenvolver uma metaheurística GRASP híbrida por meio da introdução de memória e aprendizado utilizando-se técnicas de Mineração de Dados, apresentadas no próximo capítulo.

Algoritmo 3 *AlgoritmoBuscaLocal* (*Solução*)

```
1: Seleciona aleatoriamente a ordem das vizinhanças a serem analisadas;
2: Vizinhança ← primeira vizinhança;
3: Melhora ← 1;
4: enquanto (Melhora = 1) faça
5:    N ← procuraVizinhança(Solução,Vizinhança,Melhora);
6:    se Melhora = 1 então
7:       Solução ← N;
8:    senão
9:       Vizinhança ← segunda vizinhança;
10:      N ← procuraVizinhança(Solução,Vizinhança,Melhora);
11:      se (Melhora = 1) então
12:         Solução ← N;
13:         Vizinhança ← primeira vizinhança;
14:      fim se
15:   fim se
16: fim enquanto
17: retorna Solução.
```

Capítulo 4

Mineração de Dados

Conforme dito na introdução desta dissertação, o processo de obtenção dos padrões utilizado na hibridação desenvolvida neste trabalho é aquele conhecido como Mineração de Conjuntos Freqüentes (MCF) [13]. Desta forma, este capítulo visa apresentar as técnicas e algoritmos utilizados para tal hibridação. No entanto, a MCF consiste em um subproblema da Mineração de Regras de Associação (MRA) [5, 17], sendo este, portanto, o conceito apresentado na próxima seção.

4.1 Mineração de Regras de Associação

As Regras de Associação (RA) representam um importante tipo de informação extraída em processos de Mineração de Dados (MD). As regras descrevem padrões de relacionamento entre itens de dados de um domínio específico, normalmente ocultas em grandes bases de dados de transações e que ocorrem com uma determinada freqüência nestas bases de dados. Uma aplicação típica desta técnica é a análise de bases de dados compostas por transações de compras, que consiste em identificar relacionamentos que ocorrem significativamente entre produtos comprados por clientes de um determinado estabelecimento.

Uma regra de associação é definida da seguinte forma [1]. Seja $I = \{i_1, i_2, \ldots, i_n\}$,

um conjunto de itens e seja D um conjunto de transações, onde cada transação t representa um subconjunto de itens de I, ou seja, $t \subseteq I$. Uma regra de associação R é uma implicação da forma $X \Rightarrow Y$, definida sobre I, onde $X, Y \subset I$; $X, Y \neq \emptyset$ e $X \cap Y = \emptyset$. Os conjuntos X e Y são chamados, respectivamente, antecedente e conseqüente da regra R. No contexto de análise de transações de compras, uma regra como esta pode ser lida da seguinte forma: "se um consumidor compra os itens do conjunto X, então, com um certo grau de certeza, compra também os itens do conjunto Y".

Por exemplo, a regra representada por $\{feijão\} \Rightarrow \{arroz\}$, que indica a afirmação "um cliente que compra feijão em geral compra também arroz", certamente seria extraída em uma base de dados transacional de um supermercado brasileiro, já que este é um padrão evidente nas compras da população do país.

Diz-se que um conjunto de itens tem suporte s se $s\%$ das transações na base de dados D contêm os itens que compõem este conjunto. Da mesma forma, diz-se que uma regra R, $X \Rightarrow Y$, tem suporte s se $s\%$ das transações na base de dados D contêm $X \cup Y$ e diz-se que a regra R tem confiança c se $c\%$ das transações de D que contêm o conjunto X também contêm o conjunto Y.

O processo de Mineração de Regras de Associação recebe como parâmetros definidos pelo usuário um valor mínimo para a medida de suporte ($supmin$) e um valor mínimo para a medida de confiança ($confmin$) e é subdividido em duas etapas. A primeira é chamada de Mineração de Conjuntos Freqüentes (MCF). Um conjunto freqüente é um conjunto de itens que ocorrem em pelo menos $supmin\%$ das transações da base de dados. A etapa MCF consiste em identificar todos os conjuntos que atendem a esta característica dentro da base de dados. Um exemplo ilustrando o conceito de conjunto freqüente é dado a seguir.

Seja a base de dados hipotética da Figura 4.1, composta por transações contendo itens pertencentes ao domínio $I = \{0,\ 1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ 10\}$ e seja também o valor mínimo de suporte $supmin = 50\%$. Pode-se dizer então que o conjunto $F = \{1,\ 5,\ 8\}$ é freqüente, pois ocorre em três das cinco transações da base de dados, isto é, apresenta suporte $s = 60\%$, que é maior ou igual ao valor mínimo estabelecido.

Da mesma maneira, os conjuntos $F_1 = \{1\}$, $F_2 = \{5,\ 8\}$ e $F_3 = \{9\}$, dentre outros, são também freqüentes, por ocorrerem em pelo menos 50% das transações da base de dados. Conforme dito anteriormente, um algoritmo de MCF obteria todos os subconjuntos de I que ocorram em pelo menos $supmin\%$ das transações da base de dados.

$\{1,\ 4,\ 5,\ 6,\ 7,\ 9\}$
$\{1,\ 3,\ 5,\ 8,\ 9,\ 10\}$
$\{0,\ 1,\ 2,\ 5,\ 8,\ 9\}$
$\{1,\ 2,\ 5,\ 8,\ 9,\ 10\}$
$\{0,\ 1,\ 4,\ 5,\ 7,\ 9\}$

Figura 4.1: Base de Dados Transacional

A segunda etapa do processo de Mineração de Regras de Associação consiste em gerar, para cada conjunto freqüente F obtido pela etapa anterior, regras do tipo $A \Rightarrow B$ que possuam a medida de confiança maior ou igual a $confmin\%$, tal que $A \subset F$, $B \subset F$ e $A \cup B = F$.

A etapa de MCF demanda muito mais esforço computacional quando confrontada com a segunda etapa, de geração de regras, e diversas estratégias têm sido continuamente desenvolvidas e melhoradas com o intuito de apresentar soluções eficientes para a mesma [13]. A próxima seção traz um estudo sobre a etapa de Mineração de Conjuntos Freqüentes, uma vez que este foi o processo escolhido para obtenção de padrões utilizados na hibridação apresentada neste trabalho.

4.2 Mineração de Conjuntos Freqüentes

O problema relacionado à MCF foi introduzido pela primeira vez em 1993 por Agrawal *et al.* [1], quando foi proposto o algoritmo *AIS*. A fim de evitar avaliar todos os possíveis 2^n subconjuntos de $I = \{i_1, i_2, \ldots, i_n\}$, o algoritmo *AIS* considera somente os conjuntos de k itens que ocorrem em alguma transação na base de dados.

Em 1994, o algoritmo iterativo *Apriori* [2] foi proposto, trazendo melhorias sig-

nificativas sobre a primeira estratégia. A técnica se destacou por reduzir o espaço de busca da extração de conjuntos freqüentes levando em consideração a seguinte propriedade: todo conjunto de itens que contém um subconjunto não freqüente também não é freqüente.

A cada iteração k, o algoritmo *Apriori* constrói, a partir dos conjuntos freqüentes de tamanho $k-1$, obtidos na iteração anterior, conjuntos de itens de tamanho k que terão sua freqüência verificada através de uma leitura da base de dados. Na primeira iteração, é feita uma varredura na base de dados identificando-se todos os conjuntos compostos por apenas um item e cujos suportes são maiores ou iguais a *supmin*. Os conjuntos cujos suportes são inferiores a *supmin* são desconsiderados. Para cada iteração $k \geq 2$, os passos a seguir são seguidos.

a. Os possíveis conjuntos freqüentes de tamanho k, por sua vez agrupados em um conjunto C_k, chamado conjunto de candidatos de tamanho k, são gerados a partir da combinação dos conjuntos freqüentes de tamanho $(k-1)$, encontrados durante a iteração anterior. Os candidatos de C_k são então armazenados em uma árvore *hash*. Se C_k for vazio, o algoritmo é encerrado.

b. C_k é então podado a fim de eliminar candidatos com pelo menos um subconjunto de tamanho $(k-1)$ que não seja freqüente pois, de acordo com a propriedade descrita anteriormente, estes conjuntos também não serão freqüentes;

c. A base de dados D é lida e, para cada uma de suas transações t, o suporte de cada candidato de C_k contido em t é incrementado. Após terminada a leitura de toda a base D, são identificados em C_k os conjuntos de itens cujos suportes são maiores ou iguais a *supmin*. Estes conjuntos comporão F_k, chamado conjunto dos conjuntos de itens freqüentes de tamanho k.

Durante os últimos dez anos, diversos algoritmos foram propostos para desempenhar de forma eficiente a tarefa de MCF. Muitas das estratégias propostas têm como base o algoritmo *Apriori* e aprimoraram seu desempenho através da redução de alguns dos seus fatores críticos tais como: o número de varreduras na base de

dados [33]; o custo computacional da fase de contagem de suporte [4, 23, 25, 28]; assim como o número e o tamanho das transações a serem processadas [23, 25, 28].

Os algoritmos *DCI* (*Direct Count & Intersect*) proposto em [25] e sua versão mais recente chamada *kDCI++*, proposta em [23], apresentam propostas baseadas no *Apriori*. O segundo é considerado, juntamente com o algoritmo *PatriciaMine* [29], como o estado da arte para a extração de conjuntos freqüentes [13]. No entanto, por ter uma estrutura mais simples, mas ainda assim ser um algoritmo eficiente, o primeiro foi escolhido para validar algumas das estratégias de hibridação da metaheurística GRASP com técnicas de MD, descritas no próximo capítulo. Além do *DCI*, foi também adotado o algoritmo *FPmax** [15] para a extração de Conjuntos Freqüentes Maximais, utilizados em uma das estratégias de hibridação.

Um conjunto de itens é freqüente maximal quando, além de ocorrer em pelo menos *supmin*% das transações da base de dados, isto é, ser freqüente, não possui nenhum superconjunto que também seja freqüente. Em outras palavras, um conjunto é Freqüente Maximal se, além de ser freqüente, não é subconjunto de nenhum outro conjunto freqüente.

Vale observar ainda aqui que em ambos conjuntos freqüente e freqüente maximal a ordenação dos itens dentro do conjunto não é relevante. Esta característica viabilizou o emprego dos conjuntos freqüentes e freqüentes maximais na hibridação da metaheurística GRASP com Mineração de Dados aplicada ao PEC, descritos neste capítulo, uma vez que, nas soluções obtidas para o problema abordado, a ordenação dos itens também não é relevante.

4.2.1 O Algoritmo *DCI*

Durante suas primeiras iterações, o algoritmo *DCI* utiliza as idéias apresentadas no algoritmo *DCP* (*Direct Count & Prune transactions*) [24]. Este utiliza estruturas de dados que permitem acesso direto aos conjuntos candidatos e processos que reduzem a base de dados gradualmente durante a sua execução a fim de reduzir o tempo de leitura da mesma. Esta redução da base de dados é inspirada nos procedimentos

de poda introduzidos pelo algoritmo *DHP* em [28].

Quando a base de dados, reduzida sucessivamente pelas podas efetuadas nas primeiras iterações, está pequena o bastante para ser carregada em memória principal, o algoritmo *DCI* gera uma representação verticalizada da mesma. Representar uma base verticalmente, neste caso, consiste em utilizar uma estrutura de dados composta por um conjunto de n listas de bits de tamanho m, onde n é o número de itens e m é o número de transações da base de dados. Em outras palavras, a estrutura pode ser vista como uma matriz binária de n linhas por m colunas, onde cada linha está associada a um item da base de dados e cada coluna a uma transação da mesma. Assim, se o j-ésimo bit da i-ésima lista de bits possuir o valor 1, significa que o item i está presente na transação j. De forma análoga, se seu valor for 0, significa que o item i não está presente na transação j.

Uma vez que a base de dados está verticalizada e carregada em memória principal, a contagem do suporte dos conjuntos candidatos é feita computando-se as interseções das listas de bits referentes aos itens que compõem estes conjuntos candidatos.

Segundo Han *et al.* em [18], os algoritmos similares ao *Apriori* têm como ponto negativo a geração exagerada de conjuntos candidatos, especialmente quando o suporte mínimo considerado é baixo. Esta característica consiste no principal custo computacional desta classe de algoritmos. Há uma outra classe importante de algoritmos para a tarefa de MCF que extrai os conjuntos freqüentes sem a geração de conjuntos candidatos, evitando assim este custo computacional. Pertencem a esta classe os algoritmos *PatriciaMine* [29], *FP-growth* [18] e *FPgrowth** [15], cuja variação *FPmax** [15] também é utilizada nas hibridações descritas no próximo capítulo.

4.2.2 O Algoritmo *FPmax**

Adotando uma abordagem que representa a base de dados de forma compactada, através de uma árvore denominada *FP-tree*, os algoritmos *FP-growth* e *FPgrowth** evitam a geração de conjuntos de itens candidatos. Esta árvore é acompanhada de

uma estrutura de dados auxiliar que tem como objetivo facilitar os percursos na *FP-tree*. Esta estrutura é denominada tabela de cabeçalhos (*header-table*). A *FP-tree* é construída a partir de duas leituras da base de dados e os conjuntos freqüentes são obtidos a partir de projeções da base de dados e suas respectivas representações como *FP-trees*.

O algoritmo *FPgrowth** visa reduzir a quantidade de percursos realizados pelo *FP-growth* sobre a *FP-tree*, utilizando *arrays* como estruturas de dados. O algoritmo *FPmax** é uma variação deste algoritmo para a obtenção de Conjuntos Freqüentes Maximais.

As estratégias de hibridação descritas no capítulo a seguir utilizam os algoritmos *DCI* e *FPmax** para a extração de conjuntos de itens freqüentes e freqüentes maximais, respectivamente, a partir de bases de dados transacionais compostas pelas melhores soluções obtidas ao final de diversas iterações GRASP. Neste caso, cada solução é um conjunto de itens e corresponde a uma transação. O intuito de aplicar a tarefa de MCF sobre os subótimos encontrados pela metaheurística é obter padrões característicos de soluções de boa qualidade.

Capítulo 5

A Estratégia Híbrida GRASP-MD

Neste capítulo, apresenta-se a contribuição central deste trabalho: a versão híbrida da metaheurística GRASP que incorpora técnicas de Mineração de Dados (GRASP-MD). Esta metaheurística híbrida é composta basicamente de duas fases, descritas a seguir.

A primeira fase consiste na geração de padrões encontrados freqüentemente em soluções de boa qualidade. Por sua vez, esta é dividida em duas etapas. Inicialmente, iterações do GRASP para o PEC descrito no terceiro capítulo desta dissertação são executadas durante t segundos de CPU, gerando um conjunto S de soluções distintas. Deste conjunto, é extraído um subconjunto D, denominado *conjunto elite*, composto pelas d soluções de melhor qualidade dentre aquelas que compõem S. Em seguida, um algoritmo de Mineração de Conjuntos Freqüentes (MCF) é executado sobre o conjunto elite a fim de extrair conjuntos de itens que aparecem com freqüência nestas soluções. A partir deste ponto do texto, os conjuntos de itens freqüentes obtidos pelos algoritmos de MCF serão denominados *padrões*.

A segunda fase consiste na execução, durante os mesmos t segundos de CPU da primeira fase, de iterações de uma versão adaptada do algoritmo GRASP. O procedimento de construção desta versão modificada recebe um padrão como entrada e fixa-o na solução construída de maneira que esta contenha obrigatoriamente os

itens pertencentes ao padrão. O procedimento de busca local aplicado é idêntico ao do GRASP da primeira fase, podendo acontecer, inclusive, de elementos pertencentes ao padrão serem escolhidos para serem retirados da solução nas trocas k-p.

As iterações GRASP executadas na segunda fase da versão híbrida podem trabalhar com um número variável de padrões. Em cada iteração, é utilizado um único padrão, mas diferentes padrões podem ser utilizados em diferentes iterações. O número e a forma de escolha dos padrões que serão utilizados seguem uma estratégia pré-estabelecida. Neste trabalho, foram empregadas quatro diferentes estratégias para a formação do conjunto de padrões a serem utilizados na segunda fase. A utilização de cada estratégia dá origem a uma versão diferente do algoritmo híbrido GRASP-MD. Cada uma destas versões é apresentada mais adiante.

No Algoritmo 4, é apresentado o pseudo-código do procedimento de construção do GRASP para o PEC, adaptado para trabalhar com um padrão. O parâmetro p é o padrão utilizado na iteração em questão, que guia a construção da solução. Nas linhas 1 e 2, a solução inicial é definida com todos os elementos de p. Os elementos de p e todos os itens afetados pelas restrições que os contêm são removidos do conjunto de candidatos nas linhas 5 e 6. Nas linhas de 7 a 18 é aplicado o mesmo procedimento apresentado nas linhas de 5 a 12 do Algoritmo 2 para se obter os demais elementos componentes da solução. Por fim, a solução construída, com todos os elementos de p e mais aqueles escolhidos pela execução das linhas 7 a 18, é retornada na linha 19.

Em suma, a metaheurística híbrida GRASP-MD é composta pelas seguintes etapas: (a) execução de um conjunto de iterações GRASP; (b) extração de padrões; (c) seleção de padrões e (d) execução de iterações do algoritmo GRASP adaptado, utilizando-se os padrões selecionados. As três primeiras etapas compõem a primeira fase do GRASP-MD, ou Geração do Conjunto de Padrões (GCP), enquanto a última etapa compõe a segunda fase do GRASP-MD, ou Fase Híbrida (FH).

Conforme dito anteriormente, quatro diferentes estratégias foram utilizadas para a obtenção e a seleção de padrões para serem utilizados na fase híbrida do GRASP-MD. Nas seções subseqüentes, cada uma delas é apresentada.

Algoritmo 4 *AlgoritmoConstruçãoAdaptada* (p)

1: $x_i \leftarrow 0, \forall i \in I$;
2: $x_i \leftarrow 1, \forall i \in p$;
3: $I_{it} \leftarrow I \backslash p$;
4: $I_{it} \leftarrow I_{it} \backslash \{i : \exists j \in J, \exists k \in p, t_{i,j} + t_{k,j} > 1\}$;
5: $e(i) \leftarrow c_i / \sum_{j \in J} t_{i,j}, \forall i \in I_{it}$;
6: $I_{it} \leftarrow Ordena(I_{it}, e(i))$;
7: **enquanto** $(I_{it} \neq \emptyset)$ **faça**
8: **se** $\alpha > 0$ **então**
9: $l \leftarrow \lceil \alpha \times n_{it} \rceil$
10: **senão**
11: $l \leftarrow 1$;
12: **fim se**
13: $LRC \leftarrow l$ primeiro elementos de I_{it};
14: $i^* \leftarrow escolhaAleatória(LRC)$;
15: $x_{i^*} \leftarrow 1$;
16: $I_{it} \leftarrow I_{it} \backslash \{i^*\}$;
17: $I_{it} \leftarrow I_{it} \backslash \{i : \exists j \in J, t_{i,j} + t_{i^*,j} > 1\}$;
18: **fim enquanto**
19: **retorna** X.

5.1 A Estratégia do Maior Padrão (MP)

A importância desta primeira versão híbrida do GRASP, denominada Estratégia do Maior Padrão (MP), está no fato de ter sido a primeira a ser testada e sua simplicidade se justifica por ser uma investigação inicial para corroborar a suspeita de que a introdução de MD no GRASP poderia trazer bons resultados. A partir da comprovação do potencial deste tipo de hibridação, através dos bons resultados obtidos na MP, apresentados no próximo capítulo, foi possível dar seqüência aos estudos e elaborar estratégias mais sofisticadas enfatizando diferentes aspectos da utilização de padrões. Cada uma destas estratégias será apresentada nas seções seguintes.

A MP utiliza o algoritmo *DCI* para a tarefa de extração de padrões e apenas um padrão é selecionado para ser utilizado na Fase Híbrida. A escolha desse padrão ocorre da seguinte forma. Um conjunto F de padrões com suporte maior ou igual a um valor s predefinido é obtido pelo algoritmo *DCI* a partir do conjunto elite gerado pela execução do GRASP na fase de Geração do Conjunto de Padrões. O tamanho do conjunto elite é um parâmetro de entrada do algoritmo.

De F, então, é extraído o padrão com o maior número de itens, isto é, o padrão de maior tamanho. Caso haja mais de um padrão com o maior tamanho, o padrão de maior suporte entre estes é selecionado, pois tudo indica que aquele que possui o maior suporte representa uma característica mais forte entre soluções de boa qualidade por ser compartilhado por um maior número delas. Por fim, caso haja empate ao serem considerandos tanto o tamanho quanto o suporte, a escolha se dá de forma aleatória. O padrão selecionado com este critério é utilizado em todas as iterações da Fase Híbrida do GRASP-MD.

Vale destacar aqui que esta estratégia é bastante simples por utilizar um único padrão para todas as iterações híbridas e que o critério de escolha foi o tamanho dos padrões, no intuito de utilizar o máximo de itens comuns entre as soluções do conjunto elite. Se tal abordagem simples indicasse bons resultados, seria um forte indício de que estratégias mais elaboradas que considerassem diferentes aspectos das características dos padrões poderiam trazer resultados ainda melhores. Desta forma, após a obtenção de bons resultados para a estratégia MP, baseada em um único padrão, uma nova estratégia foi concebida, enfatizando outro aspecto: a diversificação de padrões.

5.2 A Estratégia do Maior Padrão para cada Suporte (MPS)

A Estratégia do Maior Padrão para cada Suporte (MPS), diferentemente de sua predecessora, não trabalha com apenas um, mas com um conjunto de padrões. Esta estratégia utiliza a relevância destes padrões, baseando-se em seus valores de suporte na seleção do conjunto a ser utilizado. Em primeira análise, somente pela simples utilização de mais de um padrão, uma maior diversificação, se comparada com a estratégia anterior, foi introduzida. Porém, para que a seleção de padrões não fosse puramente aleatória, a idéia utilizada foi trabalhar com padrões de valores de suporte diferentes, trabalhando ora com um padrão mais relevante, ora com um menos relevante, para que se ampliasse a possibilidade de emprego de padrões com

características variadas. A seguir, é feita uma descrição da estratégia.

Assim como a MP, esta também utiliza o algoritmo *DCI* para a tarefa de extração de padrões. Porém, conforme dito anteriormente, ao invés de apenas um padrão, um conjunto P de padrões é selecionado para ser utilizado na Fase Híbrida. A escolha desse conjunto de padrões ocorre da seguinte forma. O mesmo conjunto F descrito na seção anterior é gerado para esta estratégia. De F são extraídos, para cada valor de suporte encontrado que seja maior ou igual a um valor s predefinido, o padrão que apresenta o maior tamanho (número de itens). Em caso de empate segundo o tamanho, a escolha é aleatória.

Por exemplo, considerando-se um conjunto elite com 5 soluções e supondo-se que o parâmetro s seja de duas transações, o conjunto P poderia ser composto por até quatro padrões: o maior padrão cujo suporte é 2, o maior cujo suporte é 3, o maior cujo suporte é 4 e o maior cujo suporte é 5. Destaca-se novamente que, para cada um dos valores de suporte encontrados, pode haver mais de um padrão que possa ser considerado como o de maior tamanho. Assim, nesta situação em que há empate segundo o critério do maior tamanho, dentro de um determinado valor de suporte, o padrão é selecionado aleatoriamente.

Após obtido, o conjunto P é ordenado de forma decrescente de acordo com os valores de suporte de seus padrões e estes são fornecidos, alternadamente, para as iterações da Fase Híbrida do GRASP-MD. A motivação para tal ordenação entre os padrões de P é que, caso o número de iterações realizadas nos t segundos de CPU de execução da Fase Híbrida não seja um múltiplo exato do número de padrões de P e, conseqüentemente, alguns padrões tenham que ser mais utilizados que outros, aqueles mais freqüentes tenham prioridade sobre os menos freqüentes.

Um importante aspecto desta estratégia é que, por focar a escolha dos padrões pelas características de suporte dos mesmos, ela tende a escolher padrões mais diversificados, porém menores, já que um padrão com suporte alto tende a ter poucos itens, se comparado ao conjunto completo de padrões obtidos pela técnica de extração de conjuntos freqüentes.

Focar o suporte dos padrões significa que as características mais comuns entre as soluções de elite obtidas na primeira fase serão utilizadas e a diversificação vem como conseqüência da utilização de padrões com diferentes graus de exigência, com relação à sua relevância na base de dados.

Conforme poderá ser constatado no próximo capítulo, a introdução de diversificação trouxe benefícios sobre a utilização intensiva de um único padrão na estratégia MP. O próximo passo da investigação foi estudar a contribuição combinada das características que ambas estratégias, vistas até então, focavam: o tamanho e a diversificação do uso de padrões.

5.3 A Estratégia dos n Maiores Padrões (nMP)

A terceira versão híbrida, denominada Estratégia dos n Maiores Padrões (nMP), faz uso de padrões de tamanho grande e, ao mesmo tempo, diferentes, com o intuito de trabalhar os dois aspectos focados pelas estratégias anteriores. Porém, a forma de diversificar os padrões não é a mesma da estratégia MPS, que se utiliza do suporte para guiar a seleção. Em vez disso, esta estratégia utiliza-se do tamanho dos mesmos para tentar gerar um conjunto diversificado de padrões. Esta mudança de abordagem justifica-se pela tentativa de se evitar a escolha de padrões de tamanho pequeno, uma tendência da estratégia MPS. Uma descrição desta estratégia de escolha é dada a seguir.

Assim como nas estratégias anteriores, utiliza-se o algoritmo *DCI* para a tarefa de extração de padrões. Como acontece na estratégia MPS, um conjunto P de padrões é selecionado para ser utilizado na Fase Híbrida. A escolha desse conjunto de padrões ocorre da seguinte maneira. Inicialmente, o mesmo conjunto F descrito nas estratégias anteriores é gerado. Em seguida, os n maiores padrões de F são selecionados para formar P. Em caso de empate por tamanho, os padrões com maior valor de suporte são priorizados. Em caso de empate também por suporte, a escolha é aleatória.

O conjunto P é ordenado de forma decrescente de acordo com os valores de

suporte de seus padrões, pela mesma motivação descrita anteriormente. Os padrões do conjunto P ordenado são então fornecidos alternadamente para as iterações da fase híbrida do GRASP-MD. Vale observar aqui que a estratégia MP pode ser vista como um caso particular da nMP, onde o conjunto P é composto por apenas um padrão.

Como poderá ser constatado no próximo capítulo, os resultados da utilização de padrões, levando-se em conta tanto a diversificação quanto o tamanho dos padrões trouxe melhorias para o desempenho da técnica. Porém, ainda havia o risco de, uma vez que considerava-se o tamanho dos padrões como guia para a obtenção do conjunto utilizado na fase híbrida do GRASP-MD, a estratégia levar à seleção de padrões com um alto grau de semelhança entre si, com relação aos itens que os compõem. Este aspecto em particular pode ser negativo no que diz respeito à diversificação pois, ao serem muito parecidos, dois padrões terão cargas de informação muito próximas a respeito dos itens envolvidos.

Assim, detectou-se a necessidade de se enriquecer a investigação não só utilizando as informações de tamanho e suporte dos padrões encontrados, mas também, de alguma forma, considerar a natureza dos mesmos, isto é, levar em conta os itens componentes de cada padrão e, em uma tentativa de se obter uma diversificação mais efetiva dos padrões utilizados, tentar reduzir o grau de semelhança entre os mesmos. Foi concebida então uma quarta estratégia, que levaria em conta a composição dos padrões selecionados e que tentasse reduzir as chances de uso acentuado de padrões similares.

5.4 A Estratégia dos n Maiores Padrões Maximais (nMM)

A quarta versão implementada, denominada Estratégia dos n Maiores Padrões Maximais (nMM), vale-se do uso da extração de conjuntos freqüentes maximais para reduzir a possibilidade de utilização de padrões semelhantes na fase híbrida do

GRASP-MD.

Uma vez que se garante que, considerado um mesmo valor de suporte mínimo, um conjunto freqüente maximal não é subconjunto de nenhum outro conjunto freqüente, ao se compor um conjunto de padrões, sendo todos eles freqüentes maximais, as chances de se ter padrões com níveis elevados de semelhança são reduzidas. Este fator conduz a uma diversificação mais efetiva em relação à estratégia anterior. Desta maneira, ao se trocar o uso de conjuntos freqüentes por conjuntos freqüentes maximais passou-se a levar em conta a natureza dos padrões na diversificação dos padrões empregados.

No entanto, a estratégia deveria, para continuar a se valer dos benefícios introduzidos pelas estratégias descritas anteriormente, considerar ainda as informações de tamanho e suporte dos padrões obtidos. A forma com que estes três aspectos combinados foram utilizados na concepção da estratégia nMM é descrita a seguir.

Por lidar com conjuntos freqüentes maximais, a estratégia nMM, diferentemente das estratégias anteriores, utiliza o algoritmo *FPmax** para a tarefa de extração de padrões. O conjunto P de padrões para ser utilizado na Fase Híbrida é selecionado da maneira descrita a seguir. O algoritmo *FPmax** é executado $d-1$ vezes, onde d é o número de soluções encontradas no Cojunto Elite. Cada uma das execuções utiliza um valor diferente para o suporte mínimo $s \in S = \{2, \ldots d\}$. Todos os conjuntos de padrões obtidos pelas $d-1$ execuções do *FPmax** são então combinados em um único conjunto F.

A obtenção de P, a partir de F, na estratégia nMM ocorre da mesma forma que na estratégia nMP, com os n maiores padrões de F sendo selecionados para formar P. Os critérios de desempate, a ordenação do conjunto e a seqüência de utilização dos padrões na fase híbrida do algoritmo GRASP-MD também são os mesmos da estratégia nMP.

O próximo capítulo traz a análise e os resultados experimentais da aplicação das quatro estratégias aqui descritas e da aplicação do algoritmo GRASP original a um conjunto de instâncias do PEC. Desta forma é avaliado o impacto da hibridação com

técnicas de MD na qualidade das soluções obtidas pelo GRASP, quando aplicado ao PEC. Para efeitos de consulta durante a análise dos resultados apresentados no Capítulo 6, a Tabela 5.1 traz um resumo comparativo de algumas características dos padrões empregados em cada uma das estratégias.

Tabela 5.1: Características dos padrões de cada estratégia

Estratégia	Algoritmo de MCF	Tipo de padrão utilizado	Padrões em P
MP	DCI	Freqüente	1
MPS	DCI	Freqüente	até $(d - s + 1)$
nMP	DCI	Freqüente	até n
nMM	FPmax*	Freqüente Maximal	até n

Capítulo 6

Resultados Experimentais

Neste capítulo, são apresentados os resultados obtidos a partir dos experimentos realizados com as diferentes versões do algoritmo GRASP-MD, descritas no Capítulo 5. Objetiva-se avaliar o resultado da incorporação de técnicas de Mineração de Dados na metaheurística GRASP aplicada ao PEC. Este capítulo está organizado da seguinte forma. Inicialmente, as características das instâncias utilizadas para a validação dos algoritmos são apresentadas. Em seguida, são descritos o ambiente computacional onde os testes foram realizados e os parâmetros utilizados para a execução dos algoritmos. Por fim, as estratégias híbridas são comparadas entre si e com o GRASP para o PEC descrito no Capítulo 3.

Cada uma das versões híbridas aqui apresentadas, a partir deste ponto do texto, será identificada pela sigla que referencia a estratégia de geração e seleção de padrões utilizada em sua implementação (MP, MPS, nMP ou nMM).

6.1 Instâncias Utilizadas

Foi utilizado um conjunto de quatorze instâncias apresentadas em [6]. Estas instâncias foram geradas aleatoriamente a partir dos seguintes parâmetros.

- O número de itens ($|I|$).
- O número de restrições ($|J|$).
- Percentual de elementos não nulos na matriz de restrições (D).
- Os valores de custos dos itens ($c_i, 1 \leq i \leq |I|$), uniformemente distribuídos no intervalo $[1 - 20]$.

As características das instâncias utilizadas[1] são apresentadas na Tabela 6.1, que inclui ainda o valor da melhor solução conhecida, para cada uma das instâncias empregadas na investigação. Estes valores de melhor qualidade conhecida são apresentados em [6] e foram obtidos utilizando o software *Cplex*. Vale observar que para a terceira instância, o melhor valor conhecido para uma solução desta instância (122) coincide com o valor ótimo da mesma.

Tabela 6.1: Características das Instâncias

Instância	$\|I\|$	$\|J\|$	D	Melhor valor conhecido
1	500	2500	1,23	323
2	500	2500	1,20	24
3	500	2500	2,22	122
4	500	2500	2,19	8
5	500	1500	2,17	192
6	500	1500	2,20	13
7	1000	5000	0,60	661
8	1000	5000	0,60	48
9	1000	1000	2,60	222
10	1000	1000	2,65	15
11	2000	10000	0,55	478
12	2000	10000	0,55	32
13	2000	2000	2,55	140
14	2000	2000	2,56	9

[1] *http://www.emse.fr/~delorme/Instances-uk.html*, acessada em 20/09/2005

6.2 Ambiente de Execução

Os algoritmos testados foram implementados na linguagem C, compilados com o compilador `gcc`, versão 3.2.2, no sistema operacional Linux, com *kernel* versão 2.4.20-31.9, e foram executados em um PC Pentium IV 1.7GHz, com 256 MB de memória RAM. Para todas as execuções do GRASP para o PEC realizadas, o parâmetro α, utilizado na obtenção da Lista Restrita de Candidatos, teve valor fixado em 30%, ou 0.3, enquanto o critério de parada foi de 21.600s de tempo de CPU ou 520 iterações, a condição que acontecesse em primeiro lugar, dentre estas.

Para as estratégias híbridas, tanto a fase de geração do conjunto elite quanto a fase híbrida foram executadas por 10.800s ou 260 iterações cada. Em todos os experimentos realizados, o tempo necessário para desempenhar os procedimentos de Mineração de Dados, entre as duas fases, foi inferior a um segundo, sendo assim desprezado na contagem total do tempo de execução dos algoritmos. O tamanho do conjunto elite foi definido em dez soluções, ou seja, a mineração de padrões se deu a partir das dez melhores soluções encontradas na primeira fase das versões do GRASP-MD. O suporte mínimo utilizado nos algoritmos de mineração de dados foi de duas soluções, consistindo portanto em um suporte de 20%. Desta forma, considerou-se como padrão freqüente quaisquer características compartilhadas por mais de uma solução do conjunto elite. Quanto maior o suporte do conjunto, mais forte e representativo será o padrão.

Vale ressaltar neste ponto que decidiu-se não variar tanto o parâmetro α quanto o critério de parada pois o foco da pesquisa consistia em isolar a contribuição da introdução de Mineração de Dados na metaheurística. A variação desses parâmetros consistiria em avaliar o desempenho do GRASP propriamente dito, quando aplicado ao problema. A variação do parâmetro α e sua influência no desempenho do GRASP para o PEC é estudada em [6].

Para a obtenção dos resultados apresentados, todas as estratégias foram executadas, para cada instância, dez vezes utilizando-se dez sementes diferentes. São reportados para análise: o desvio padrão, a média e o melhor valor obtido nestas

dez execuções de cada estratégia, para cada instância.

6.3 Comparação Entre as Estratégias

A ordem dos resultados apresentados nesta seção é decorrente da seqüência de desenvolvimento e das observações feitas acerca das estratégias apresentadas no capítulo anterior. Desta forma, a Tabela 6.2 compara o comportamento do GRASP para o PEC (G) com a primeira abordagem híbrida, a Estratégia do Maior Padrão (MP), que utiliza apenas um padrão em todas as iterações da fase híbrida do GRASP-MD. O objetivo é avaliar se a introdução de técnicas de MD pode contribuir no desempenho da metaheurística.

A primeira coluna da Tabela 6.2 identifica as instâncias utilizadas. A segunda e terceira colunas da tabela trazem, respectivamente, a média entre os valores encontrados pelas dez execuções do GRASP e o melhor resultado entre estes. A quarta coluna traz o desvio padrão calculado para os mesmos dez valores utilizados no cálculo da média. A quinta, sexta e sétima colunas trazem a média, melhor solução e desvio padrão para os resultados da estratégia MP. Para todas as instâncias, os melhores valores de cada critério (média, melhor valor e desvio padrão) estão destacados em negrito. Deste ponto do texto em diante, este recurso será utilizado em todas as tabelas comparativas envolvendo as estratégias implementadas, assim como a abreviação G para denotar o GRASP para o PEC.

Considerando os valores das médias para as quatorze instâncias da tabela, a estratégia híbrida MP obteve melhores resultados para onze delas, enquanto o GRASP obteve melhor resultado em apenas uma instância, havendo portanto, dois empates. Para os melhores valores, o GRASP-MD obteve melhor desempenho em quatro instâncias, enquanto o GRASP simples conseguiu superá-lo em apenas duas delas. Nas demais oito instâncias, ambas estratégias obtiveram resultados idênticos quanto a este critério. Em praticamente todas instâncias, o desvio padrão observado para ambas estratégias foi bem pequeno, o que demonstra estabilidade na qualidade do desempenho de ambas técnicas.

Tabela 6.2: Comparação entre as estratégias GRASP e híbrida MP

Instância	G			MP		
	Média	Melhor	Desvio Padrão	Média	Melhor	Desvio Padrão
1	315,2	320	5,6	**317,4**	**323**	4,7
2	23,1	24	0,7	**23,4**	24	0,5
3	119,3	122	1,7	119,3	122	1,7
4	7,7	8	0,5	**7,9**	8	0,3
5	187,5	192	4,8	187,5	192	4,8
6	12,6	13	0,5	**12,9**	13	0,3
7	589,4	604	8,0	**599,3**	**616**	13,7
8	43,3	44	0,5	**44,0**	**45**	0,8
9	**212,4**	**216**	2,3	211,0	215	2,9
10	13,1	14	0,3	**13,8**	**15**	0,6
11	409,6	**443**	14,5	**410,1**	431	13,6
12	28,1	30	0,7	**28,4**	30	0,7
13	129,0	135	3,4	**130,5**	135	2,9
14	8,4	9	0,5	**8,7**	9	0,5

A análise desta tabela evidencia a eficácia da utilização de padrões para guiar iterações GRASP durante a fase híbrida do GRASP-MD, principalmente se levado em consideração o desempenho médio de cada algoritmo. Nota-se portanto, uma sensível melhoria no desempenho global, considerando-se o conjunto completo de instâncias quando da introdução de MD na técnica tradicional.

Uma vez que a estratégia MP trouxe uma contribuição positiva sobre o GRASP, a Tabela 6.3 traz uma análise comparativa da primeira, confrontada com a Estratégia do Maior Padrão para cada Suporte (MPS), a estratégia seguinte na ordem de apresentação do capítulo anterior, que considera o maior padrão para cada valor distinto de suporte encontrado durante a etapa de Mineração de Dados. O intuito de tal análise é investigar os efeitos da introdução de diversificação dos padrões utilizados na fase híbrida da metaheurística. Os significados das colunas desta tabela são os mesmos da tabela anterior, assim como o significado dos valores destacados em negrito.

No confronto entre os desempenhos médios destas duas estratégias, a MPS apresentou melhores resultados para nove instâncias, enquanto a MP se mostrou melhor em três casos, havendo ainda dois empates. Considerando-se os melhores valores obtidos, as técnicas se comportaram de forma similar, encontrando resultados idênticos em onze instâncias. A estratégia MPS apresentou desempenho superior em apenas um caso e, nos dois casos restantes, a primeira estratégia híbrida encontrou valores melhores. Quanto ao desvio padrão, assim como na comparação da tabela anterior, os valores encontrados são pequenos, com ligeira vantagem para a MPS. Mais uma vez, tais valores demonstram estabilidade de desempenho dos algoritmos analisados.

A análise da Tabela 6.3 demonstra que, em média, o desempenho do GRASP híbrido pôde ser melhorado introduzindo-se diversificação do uso de padrões na sua fase híbrida (MPS) ao invés do uso intensivo de um único padrão (MP).

A Tabela 6.4 traz, em seguida, a comparação entre as estratégias MPS e nMP (Estratégia dos n Maiores Padrões), no intuito de investigar os efeitos da combinação dos fatores de tamanho e de suporte dos padrões simultaneamente na seleção dos

Tabela 6.3: Comparação entre as estratégias híbridas MP e MPS

Instância	MP			MPS		
	Média	Melhor	Desvio Padrão	Média	Melhor	Desvio Padrão
1	317,4	323	4,7	**320,1**	323	3,1
2	23,4	24	0,5	23,4	24	0,5
3	119,3	122	1,7	**120,2**	122	1,5
4	7,9	8	0,3	7,9	8	0,3
5	187,5	192	4,8	**191,4**	192	1,4
6	**12,9**	13	0,3	12,8	13	0,4
7	599,3	616	13,7	**601,3**	616	10,1
8	**44,0**	**45**	0,8	43,5	44	0,5
9	211,0	215	2,9	**213,8**	**222**	3,3
10	13,8	15	0,6	**13,9**	15	1,0
11	410,1	**431**	13,6	**414,1**	428	11,4
12	**28,4**	30	0,7	28,1	30	0,9
13	130,5	135	2,9	**131,2**	135	3,1
14	8,7	9	0,5	**8,8**	9	0,4

mesmos para a fase híbrida do GRASP-MD. Como a estratégia MPS demonstrou melhoria global sobre a estratégia MP, que por sua vez demonstrou melhoria global sobre o GRASP para o PEC, somente a primeira será confrontada com o método nMP nesta análise. A estratégia nMP utiliza os n maiores padrões encontrados durante a etapa de Mineração de Dados. Para este experimento, foi utilizado $n = 10$ e a estrutura da tabela é análoga à estrutura das duas anteriores.

Tabela 6.4: Comparação entre as estratégias híbridas MPS e nMP

Instância	MPS			nMP		
	Média	Melhor	Desvio Padrão	Média	Melhor	Desvio Padrão
1	**320,1**	323	3,1	318,9	323	3,1
2	23,4	24	0,5	23,4	24	0,5
3	120,2	122	1,5	**120,8**	122	1,4
4	7,9	8	0,3	7,9	8	0,3
5	**191,4**	192	1,4	188,0	192	5,0
6	12,8	13	0,4	**12,9**	13	0,3
7	**601,3**	616	10,1	600,1	**624**	10,8
8	43,5	44	0,5	**44,3**	**46**	0,8
9	213,8	222	3,3	**214,4**	222	3,2
10	13,9	15	1,0	**14,1**	15	0,9
11	**414,1**	428	11,4	409,6	**431**	11,9
12	28,1	30	0,9	**28,2**	30	0,9
13	131,2	135	3,1	131,2	135	3,3
14	8,8	9	0,4	**8,9**	9	0,3

O método de seleção de padrões nMP trouxe para o GRASP-MD uma ligeira melhoria de desempenho médio, pois em sete das quatorze instâncias, esta estratégia obteve médias melhores de resultados, sendo superada pelo método anterior em três casos, havendo ainda três empates. Quanto aos melhores resultados obtidos, as técnicas mostraram desempenho bastante semelhantes, tendo obtido resultados iguais em onze instâncias. Nas três instâncias restantes, a estratégia nMP obteve desempenho melhor neste critério. Pode-se notar que os valores de desvio padrão

de ambas estratégias são bastante próximos em praticamente todos os casos, sendo todos eles pequenos. Isto evidencia que a estabilidade de ambas as técnicas é similar.

Este terceiro experimento demonstra que levar em consideração os fatores tamanho e suporte simultaneamente durante o uso de padrões na fase híbrida do GRASP-MD trouxe melhorias para a metaheurística.

A Tabela 6.5, em seguida, traz a comparação do comportamento das estratégias nMP e nMM, que considera os n maiores padrões maximais, com valores de suporte mínimo variando de dois a cinco. Em ambas estratégias, foi utilizado o valor $n = 10$ e a estrutura é análoga às das Tabelas 6.2, 6.3 e 6.4. A estratégia nMM obteve melhores resultados médios em nove instâncias, enquanto a estratégia nMP obteve melhor resultado em quatro, havendo ainda um empate. Quanto aos melhores valores encontrados, estes foram os mesmos em doze instâncias e cada estratégia superou a outra uma vez, demonstrando um equilíbrio no desempenho das duas estratégias nesse quesito. Na maioria dos casos, a estratégia nMM obteve menor desvio padrão do que a nMP, demonstrando ser mais robusta que a primeira.

Este quarto experimento evidencia que a estratégia nMM obteve melhor desempenho do que a estratégia nMP. Isto é, a introdução de mais diversidade nos padrões selecionados, através da utilização de padrões maximais, trouxe melhorias para a técnica híbrida.

A análise das tabelas comparativas entre cada par de estratégias evidencia que aquela com melhor desempenho é a nMM. Esta conclusão pode ser reforçada pela análise da Tabela 6.6. Para cada uma das instâncias, são apresentados os percentuais do valor médio encontrado pelo GRASP simples (coluna G) e por cada estratégia híbrida (colunas MP, MPS, nMP e nMM) em relação à melhor solução conhecida. Na penúltima linha, são apresentados as médias destes valores percentuais, indicando o desempenho médio dos algoritmos sobre todo o conjunto de instâncias. Por fim, na última linha, é trazido o ganho médio percentual de cada implementação sobre o GRASP simples, considerando também o conjunto completo de instâncias. Assim como nas tabelas anteriores, os melhores resultados de cada linha estão destacados em negrito. Como pode ser observado, a estratégia nMM apresenta o melhor de-

Tabela 6.5: Comparação entre as estratégias híbridas nMP e nMM

Instância	nMP			nMM		
	Média	Melhor	Desvio Padrão	Média	Melhor	Desvio Padrão
1	318,9	323	3,1	**321,2**	323	1,9
2	23,4	24	0,5	**23,5**	24	0,5
3	**120,8**	122	1,4	120,0	122	1,3
4	7,9	8	0,3	7,9	8	0,3
5	188,0	192	5,0	**191,6**	192	1,3
6	12,9	13	0,3	**13,0**	13	0,0
7	600,1	**624**	10,8	**605,1**	614	7,1
8	**44,3**	46	0,8	44,0	46	1,1
9	214,4	222	3,2	**216,0**	222	3,4
10	**14,1**	15	0,9	14,0	15	0,7
11	409,6	431	11,9	**417,3**	**439**	11,1
12	28,2	30	0,9	**28,4**	30	0,8
13	131,2	135	3,3	**131,5**	135	2,5
14	**8,9**	9	0,3	8,5	9	0,5

sempenho em nove das quatorze instâncias, além de possuir o melhor valor médio para o conjunto completo e trazer o maior ganho médio percentual sobre o GRASP.

Tabela 6.6: Percentual médio em relação ao melhor valor conhecido

Instância	G	MP	MPS	nMP	nMM
1	97,59	98,27	99,10	98,73	**99,44**
2	96,25	97,50	97,50	97,50	**97,92**
3	97,79	97,79	98,52	**99,02**	98,36
4	96,25	98,75	98,75	98,75	98,75
5	97,66	97,66	99,69	97,92	**99,79**
6	96,92	99,23	98,46	99,23	**100,0**0
7	89,17	90,67	90,97	90,79	**91,54**
8	90,21	91,67	90,63	**92,29**	91,67
9	95,68	95,05	96,31	96,58	**97,30**
10	87,33	92,00	92,67	**94,00**	93,33
11	85,69	85,79	86,63	85,69	**87,30**
12	87,81	88,75	87,81	88,13	**88,75**
13	92,14	93,21	93,71	93,71	**93,93**
14	93,33	96,67	97,78	**98,89**	94,44
Média	93,13	94,50	94,90	95,09	**95,18**
Ganho médio sobre G	–	1,37	1,77	1,96	**2,05**

Ainda no intuito de estudar as melhorias de cada estratégia híbrida sobre o GRASP, o gráfico da Figura 6.1 traz o número de vezes em que cada estratégia obteve melhores resultados que o GRASP (barra cinza), o número de vezes que obteve resultados idênticos aos do GRASP (barra mais escura) e o número de vezes que apresentou resultados inferiores aos do GRASP (barra mais clara). A análise de freqüência apresentada no gráfico foi obtida a partir das dez diferentes execuções de cada implementação para cada uma das quatorze instâncias, totalizando 140 resultados por estratégia. Para cada semente utilizada, os resultados dos algoritmos de cada versão do GRASP-MD foi confrontado com o resultado do GRASP para a mesma semente e uma ocorrência era contabilizada para a freqüência da coluna

correspondente ao resultado da comparação (melhor, igual ou inferior ao GRASP). Pode ser observado que a estratégia nMM superou o desempenho das demais propostas híbridas uma vez que a mesma obteve uma freqüência maior de melhores resultados com relação ao GRASP e também obteve freqüência menor de resultados piores com relação ao GRASP. A análise do gráfico demonstra ainda que a estratégia nMM foi a única em que a maior freqüência foi a de resultados que superam o algoritmo convencional. Nas demais, a maior freqüência é a de resultados de valores iguais. Portanto, mais uma vez a análise dos resultados leva à conclusão que a estratégia de melhor desempenho é a nMM.

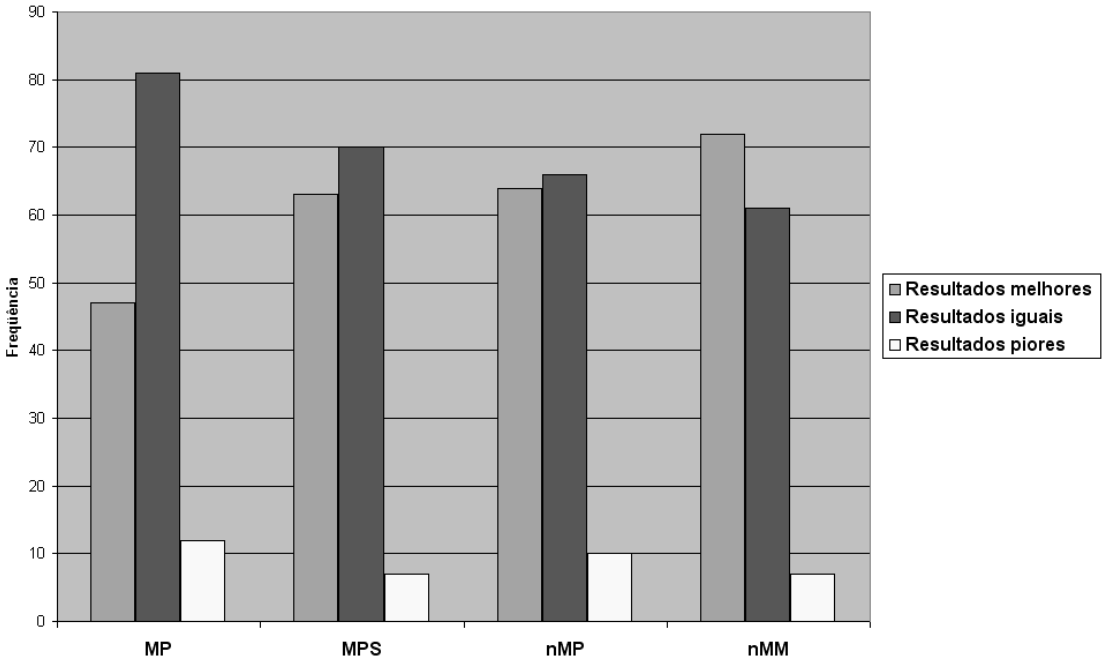

Figura 6.1: Análise de freqüência das estratégias híbridas.

O gráfico da Figura 6.2 traz a análise de freqüência dos erros da estratégia híbrida nMM e do algoritmo convencional, com relação aos melhores valores conhecidos de soluções para cada instância. Assim como no gráfico anterior, a análise de freqüência apresentada foi obtida a partir das dez diferentes execuções de cada implementação para cada uma das quatorze instâncias, totalizando 140 resultados por estratégia. Cada barra representa a freqüência de diferentes faixas de erros percentuais para cada uma das implementações. Por exemplo, a primeira barra indica o número de vezes em que o algoritmo obteve uma solução com qualidade inferior à melhor

solução conhecida, com a diferença percentual entre estas variando de 0% a 5%, isto é, o número de vezes que foram obtidas soluções com valores de 95% a 100% do melhor valor conhecido para uma instância. Todas as soluções obtidas pelas 140 execuções de cada algoritmo apresentaram desvio menores ou iguais a 20% dos melhores valores conhecidos para cada instância. A análise do gráfico permite observar que a estratégia híbrida nMM obteve mais soluções com erros pequenos (0% a 5% e 6% a 10%) e menos soluções com erros grandes (11% a 15% e 16% a 20%) que o algoritmo GRASP convencional.

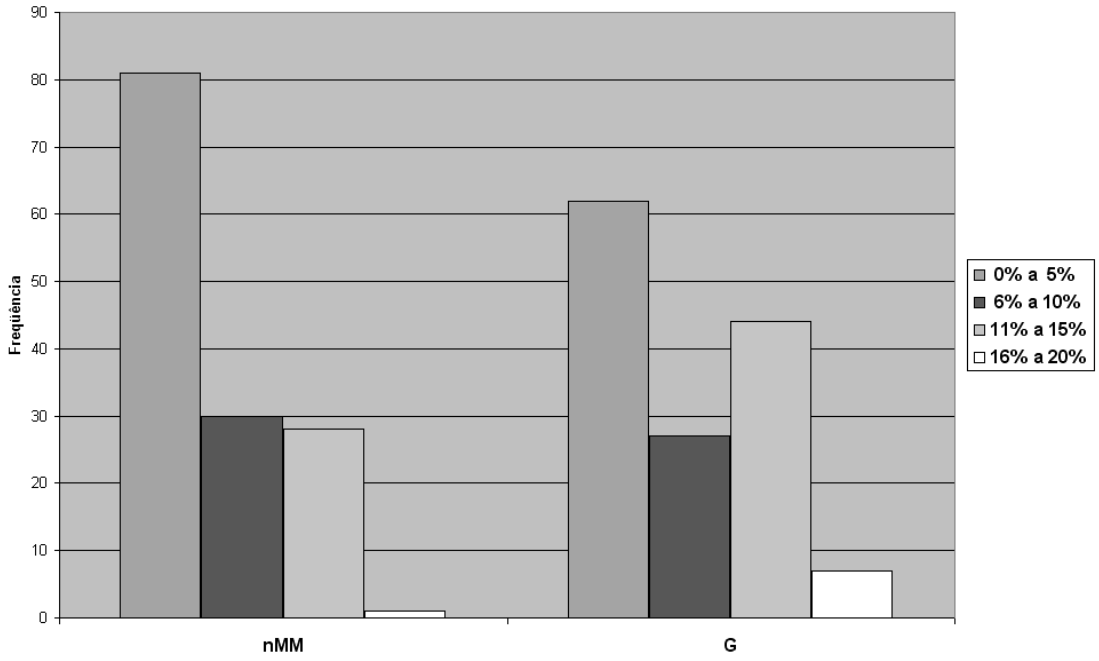

Figura 6.2: Análise de freqüência dos algoritmos nMM e GRASP.

A Tabela 6.7, a seguir, compara o melhor valor obtido pelo GRASP convencional (coluna G) e por cada versão do GRASP-MD (colunas MP, MPS, nMP e nMM), com o melhor valor conhecido, para cada instância. As colunas apresentam a diferença percentual entre os valores, isto é, o quanto a melhor solução de cada estratégia foi inferior à melhor solução conhecida, em termos percentuais. Os resultados demonstram que todas as estratégias híbridas encontraram a melhor solução conhecida um número superior de vezes do que o algoritmo convencional, o que evidencia que em termos globais a hibridação trouxe melhorias à técnica. As estratégias MPS, nMP e nMM encontraram os melhores valores (representados pelo valor $0,0$) nove vezes

entre as quatorze instâncias. Nas cinco demais instâncias os resultados foram os seguintes. Em duas delas, as versões MPS, nMP e nMM obtiveram o melhor desempenho, apresentando valores idênticos; em uma, o algoritmo nMP obteve melhor resultado; em outra, houve empate entre as estratégias nMP e nMM e, por fim, na última dessas cinco instâncias, o melhor desempenho coube ao algoritmo GRASP.

Tabela 6.7: Diferença percentual entre o melhor valor obtido por cada estratégia e o melhor valor conhecido

Instância	G	MP	MPS	nMP	nMM
1	0,93	0,0	0,0	0,0	0,0
2	0,0	0,0	0,0	0,0	0,0
3	0,0	0,0	0,0	0,0	0,0
4	0,0	0,0	0,0	0,0	0,0
5	0,0	0,0	0,0	0,0	0,0
6	0,0	0,0	0,0	0,0	0,0
7	8,62	6,81	6,81	5,60	7,11
8	8,33	6,25	8,33	4,17	4,17
9	2,70	3,15	0,0	0,0	0,0
10	6,67	0,0	0,0	0,0	0,0
11	7,32	9,83	10,46	9,83	8,16
12	6,25	6,25	6,25	6,25	6,25
13	3,57	3,57	3,57	3,57	3,57
14	0,0	0,0	0,0	0,0	0,0

Os experimentos a seguir tentam evidenciar que a estratégia híbrida nMM, escolhida aqui por ter apresentado um melhor desempenho entre as demais propostas, pode atingir um valor específico de qualidade de uma solução mais rapidamente que o algoritmo GRASP convencional. Para tal, ambas implementações foram executadas 120 vezes, com 120 sementes diferentes, até que uma solução com qualidade igual ou superior ao alvo fosse atingida. Foram escolhidos dois alvos distintos, um fácil e um difícil, para serem testados com a instância identificada pelo número 9. Os valores para os alvos fácil e difícil são, respectivamente, 213 e 216. A escolha desta

instância se deve ao fato de a mesma apresentar, comparativamente, baixo custo computacional, dentre as instâncias maiores, com 1000 e 2000 itens. As Figuras 6.3 e 6.4 mostram, para cada um dos alvos, a avaliação de ambos algoritmos. O eixo das abscissas traz as sementes utilizadas, numeradas de 1 a 120, enquanto o eixo da ordenadas traz o instante, em segundos, na qual o alvo é atingido ou ultrapassado. Pode-se observar que em quase todas as execuções, para ambos os alvos, a estratégia híbrida nMM alcançou o alvo antes do algoritmo GRASP. Nota-se também que para o alvo mais difícil, o algoritmo híbrido foi ainda mais eficiente.

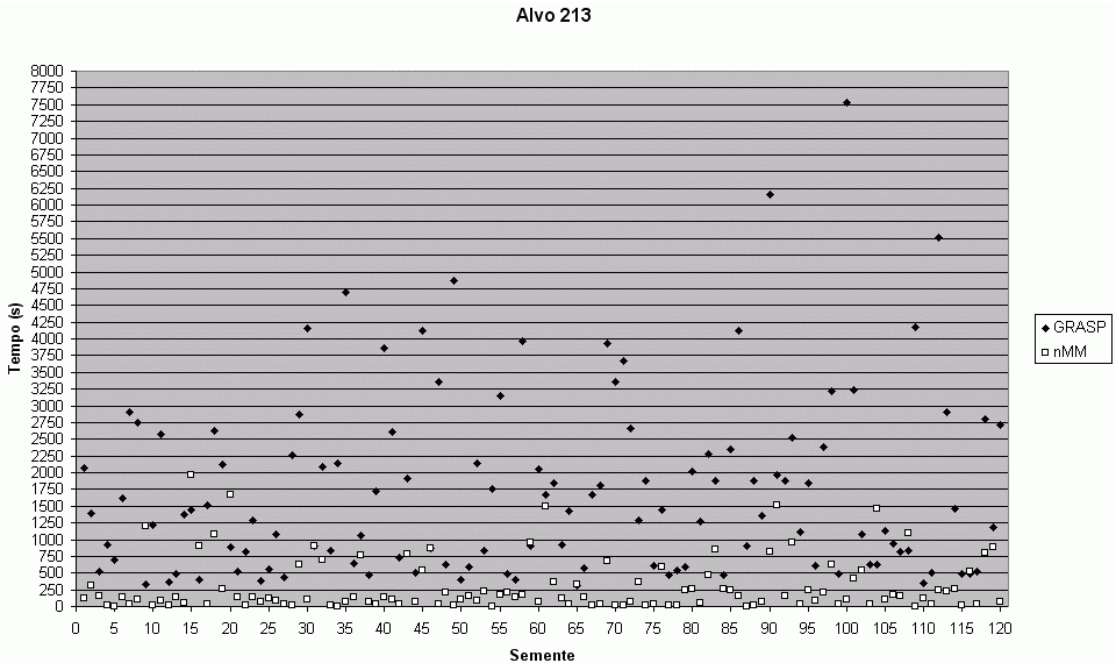

Figura 6.3: Alvo fácil para a instância 9.

A análise que se segue tem como objetivo avaliar a utilização de um conjunto elite menor. Uma vez que a estratégia nMM apresentou o melhor desempenho global, o experimento descrito pela Tabela 6.8 considera somente esta estratégia. A estratégia nMM foi executada com dez sementes diferentes para cada uma das instâncias do conjunto trabalhado. Neste experimento, as cinco melhores soluções encontradas na primeira fase do GRASP-MD compuseram o conjunto elite a ser minerado. Os resultados destas execuções são confrontados com os resultados obtidos pela mesma estratégia, utilizando dez soluções no conjunto elite. Conforme pode ser observado na Tabela 6.8, a versão utilizando elite de tamanho dez obteve desempenho superior

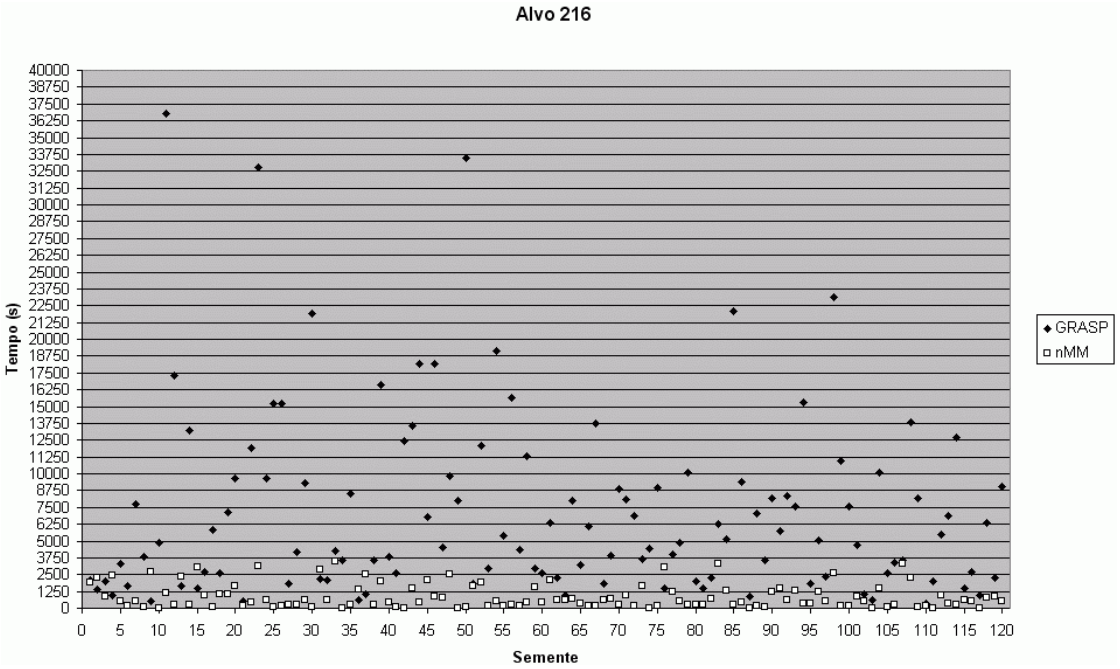

Figura 6.4: Alvo difícil para a instância 9.

em nove das quatorze instâncias, enquanto nas cinco instâncias restantes a diferença dos resultados médios a favor da versão que utiliza cinco soluções no conjunto elite foi muito pequena em quatro dos casos. Desta forma, melhorias globais não foram detectadas na técnica após a utilização de um conjunto de padrões mais "elitista".

Também foram analisadas algumas características dos conjuntos de padrões utilizados, no intuito de tentar identificar justificativas para os diferentes resultados obtidos pelas diferentes versões híbridas. Quatro características principais foram identificadas e estudadas neste trabalho: o tamanho dos padrões, o número de padrões utilizados por cada estratégia, o suporte dos padrões e o número de interseções entre os mesmos. No contexto deste trabalho, define-se como interseção entre dois padrões o número de itens em comum entre eles. A análise comparativa da influência de cada uma destas características sobre a qualidade da hibridação é realizada a seguir.

A Tabela 6.9, apresenta os tamanhos médios dos padrões utilizados em cada uma das estratégias, consideradas as dez execuções com sementes diferentes. Nota-se que a coluna correspondente ao algoritmo MP possui a maior média em todas instâncias, pois se trata da estratégia que escolhe o maior padrão. Pode-se notar

Tabela 6.8: Avaliação do tamanho do conjunto elite

Instância	Conjunto elite com 5 soluções	Conjunto elite com 10 soluções
1	319,8	**321,2**
2	**23,6**	23,5
3	**120,2**	120,0
4	**8,0**	7,9
5	190,2	**191,6**
6	12,8	**13,0**
7	603,9	**605,1**
8	43,5	**44,0**
9	215,6	**216,0**
10	13,8	**14,0**
11	415,7	**417,3**
12	28,0	**28,4**
13	**132,9**	131,5
14	**8,7**	8,5

também que a coluna correspondente ao algoritmo MPS possui a menor média em doze das quatorze instâncias. Esta estratégia utiliza padrões com valores de suporte diferentes, e aqueles com suporte alto tendem a ser pequenos, reduzindo a média.

Ainda pode ser observado que a estratégia nMM, que apresentou o melhor desempenho médio, possui tamanhos médios de padrões inferiores aos da estratégia nMP para todas as instâncias. Estas duas versões híbridas têm como característica comum o fato de trabalharem com um número semelhante de padrões. Assim, como a estratégia nMP prioriza padrões grandes e a nMM considera padrões maximais, a análise da Tabela 6.9 sugere que esta última característica possui maior importância na qualidade média das soluções do que o tamanho dos padrões utilizados.

Tabela 6.9: Tamanhos médios dos padrões utilizados

Instância	MP	MPS	nMP	nMM
1	12,80	4,67	11,92	7,90
2	11,00	4,28	10,15	6,63
3	5,00	2,32	3,89	2,29
4	2,90	1,83	2,02	1,46
5	9,00	3,60	8,10	4,17
6	7,60	3,21	6,63	3,61
7	11,00	4,07	10,12	6,76
8	13,30	4,65	12,42	8,05
9	7,40	2,94	6,43	3,65
10	4,80	2,56	3,83	2,63
11	4,40	2,43	3,41	2,28
12	5,40	2,79	4,34	3,04
13	2,40	1,70	1,56	1,35
14	2,90	1,93	2,00	1,42

Outra característica avaliada foi o numero médio de padrões utilizados pelas estratégias. A Tabela 6.10 traz a quantidade média de padrões utilizados em cada uma delas, consideradas as dez execuções com sementes diferentes. Nota-se que a coluna referente ao algoritmo MP apresenta todos os seus valores iguais a um em

decorrência da própria estratégia, que trabalha com apenas um padrão, o maior entre todos encontrados. Os valores da coluna correspondente à estratégia MPS apresentam uma variação maior como conseqüência da própria abordagem de seleção da estratégia. Por escolher o maior padrão para cada valor de suporte encontrado, o tamanho do conjunto de padrões utilizados está diretamente condicionado à faixa de valores de suporte dos conjuntos freqüentes minerados. Desta maneira, se o algoritmo de Mineração de Dados encontrar, por exemplo, padrões com valores de suporte variando de dois a cinco, o número de padrões selecionados pela estratégia será de apenas quatro padrões, uma para cada valor inteiro em $[2, 5]$.

Observa-se ainda que apesar de o tamanho do conjunto de padrões ter sido fixado em dez padrões para as estratégias nMP e nMM, em alguns poucos casos o número total de padrões obtidos pelos algoritmos de mineração foram ligeiramente inferiores a este valor. Portanto, nestes casos, foram utilizados todos os padrões encontrados pelas técnicas de MD, o que explica os valores médios diferentes de dez nas duas últimas colunas.

Através da análise das Tabelas 6.10 e 6.6, observa-se, aparentemente, que o desempenho médio das estratégias cresce de acordo com o número de padrões utilizados em cada uma delas, isto é, as estratégias que utilizaram em média menos padrões obtiveram resultados médios inferiores, enquanto as que utilizaram um número médio maior de padrões obtiveram melhores desempenhos. No entanto, tal análise pode estar equivocada, uma vez que as heurísticas de escolha de padrões variam entre as estratégias. Desta forma, o único par de algoritmos que apresenta a mesma estratégia de seleção de padrões é aquele formado pelas estratégias MP e nMP, que priorizam padrões de maior tamanho possível. Confrontando-se os resultados destes dois algoritmos, que se diferenciam apenas pelo número de padrões utilizados, pode-se observar que a diversificação de padrões se mostrou importante para o bom desempenho da técnica.

Vale observar ainda que por trabalhar com padrões maximais, a estratégia nMM descarta a seleção de padrões que sejam subconjuntos de outros padrões. Desta maneira, a escolha de um padrão de tamanho grande contido em outro padrão

grande não ocorre, enquanto na estratégia nMP tal escolha pode acontecer, pois a mesma se orienta exclusivamente pelo tamanho dos padrões. Esta diferença entre as estratégias pode, eventualmente, conduzir a uma redução no número de soluções de tamanho grande selecionadas pela técnica nMM. Assim, a utilização de padrões maximais pode, eventualmente, levar a uma redução no tamanho médio dos padrões, com relação à técnica nMP.

Tabela 6.10: Número médio de padrões utilizados

Instância	MP	MPS	nMP	nMM
1	1	7,3	10	10
2	1	5,3	10	10
3	1	3,8	10	9,9
4	1	2,2	10	9,9
5	1	5,3	10	10
6	1	4,5	10	10
7	1	5	10	10
8	1	5,2	10	10
9	1	4,9	10	10
10	1	3,2	10	10
11	1	2,6	10	10
12	1	2,9	10	10
13	1	2,1	8,6	7,4
14	1	2,1	9,4	7,7

A Tabela 6.11 estabelece uma comparação entre os valores médios de suporte dos padrões utilizados em cada estratégia híbrida, consideradas as dez execuções com sementes diferentes. Observa-se que as estratégias MP e nMP apresentam valores médios de suporte muito similares. Isto pode ser explicado pelo fato de que ambas utilizam padrões de maior tamanho possível, que tendem a ter valores de suporte menores. No entanto, o desempenho médio da segunda foi superior ao da primeira, o que pode ser constatado através da Tabela 6.6. A Tabela 6.9 também traz valores médios próximos, embora não tão próximos quanto os de suporte, para as duas

estratégias, no que tange ao tamanho dos padrões utilizados. Estes fatos sugerem que o principal fator a influenciar a diferença de desempenho entre as duas técnicas foi o número de padrões utilizados nas mesmas, o que reforça as observações feitas a respeito da tabela anterior. Tal observação pode ainda ser reforçada pela constatação de que apesar de trabalhar com os maiores valores de suporte, a estratégia MPS teve desempenho médio inferior aos das estratégias que utilizam um número maior de padrões.

Tabela 6.11: Suporte médio dos padrões utilizados

Instância	MP	MPS	nMP	nMM
1	2,00	4,56	2,00	2,49
2	2,00	4,19	2,00	2,34
3	2,00	3,68	2,01	2,82
4	2,10	2,60	2,05	2,24
5	2,00	3,96	2,00	2,67
6	2,00	3,80	2,00	2,53
7	2,00	4,16	2,00	2,34
8	2,00	4,16	2,00	2,46
9	2,00	3,76	2,00	2,53
10	2,00	3,20	2,00	2,32
11	2,00	2,80	2,00	2,33
12	2,00	3,01	2,00	2,15
13	2,30	2,92	2,23	2,41
14	2,10	2,65	2,12	2,27

A Tabela 6.12 traz o número médio percentual de interseções entre padrões para as estratégias híbridas MPS, nMP e nMM, consideradas as dez execuções com sementes diferentes. A estratégia MP foi excluída desta análise por utilizar um único padrão. O número médio de interseções apresentado nas células da tabela é calculado da forma descrita a seguir. Para cada execução de uma versão do GRASP-MD, são geradas todas as combinações possíveis de pares de padrões entre aqueles escolhidos pela estratégia e é computado o número de interseções entre estes pares. A média

entre os valores de interseções de todos os pares gerados é calculada, obtendo-se o valor médio do número de interseções para aquela execução. Em seguida, é calculada a média das médias de cada execução, que representa o número médio de interseções entre padrões para uma dada instância. Este valor é calculado para medir o grau de semelhança entre os padrões utilizados. Quanto menor o valor, mais diferentes são os padrões. Por fim, o resultado mostrado na tabela consiste em dividir este valor pelo tamanho médio dos padrões apresentados na Tabela 6.9, para indicar a representatividade do número de interseções perante o tamanho dos padrões. Por exemplo, o valor 39,83 na célula da primeira coluna da primeira linha da tabela indica que, em média, os padrões da primeira instância possuem um grau de semelhança de 39,83%.

A Tabela 6.12 apresenta uma discrepância da estratégia nMP em relação às demais no que diz respeito ao número percentual médio de interseções entre padrões em todos os casos, à exceção de uma instância. Da mesma forma, pode-se observar que as estratégias nMM e MPS utilizam padrões menos semelhantes entre si do que a primeira estratégia. Os algoritmos nMP e nMM apresentam-se nivelados em relação ao número de padrões utilizados. No entanto, esta última apresenta valores médios percentuais de interseção menores e suportes médios superiores que o da anterior para todas as instâncias. Esta constatação sugere que a redução do grau de similaridade dos padrões através da utilização de conjuntos freqüentes maximais e o aumento do suporte médio dos mesmos trouxe melhorias à técnica.

Este capítulo trouxe a análise e os resultados experimentais da aplicação das quatro estratégias híbridas descritas no capítulo anterior e da aplicação do algoritmo GRASP convencional a um conjunto de instâncias do PEC, com o objetivo de avaliar o impacto da hibridação com as técnicas de Mineração de Dados na qualidade das soluções obtidas pelo GRASP, quando aplicado a este problema. Desta forma as conclusões finais obtidas a partir das análises aqui descritas são apresentadas no Capítulo 7, bem como sugestões de trabalhos futuros.

Tabela 6.12: Número médio percentual de interseções entre os padrões utilizados

Instância	MPS	nMP	nMM
1	39,83	88,76	41,39
2	34,58	80,10	26,24
3	24,57	58,35	20,52
4	14,75	34,65	6,16
5	35,00	80,49	29,50
6	36,76	74,81	25,48
7	29,73	82,21	16,86
8	21,94	83,98	16,65
9	30,95	80,72	25,75
10	29,30	52,48	18,63
11	15,23	44,57	7,02
12	25,81	51,38	8,55
13	31,18	23,72	8,15
14	27,46	44,00	6,34

Capítulo 7

Conclusões Finais e Trabalhos Futuros

O objetivo deste trabalho foi investigar a utilização de padrões, extraídos através de técnicas de Mineração de Dados, que representam características de soluções subótimas de um problema de otimização combinatória, para guiar a obtenção de novas e melhores soluções. A idéia central foi, através da introdução de técnicas de Mineração de Conjuntos Freqüentes, incluir aprendizado de máquina na metaheurística GRASP visando melhorar seu desempenho.

A escolha do Problema do Empacotamento de Conjuntos para validar a proposta se deve à sua compatibilidade com a Mineração de Conjuntos Freqüentes, já que suas soluções se apresentam sob a forma de conjuntos de itens e um conjunto de soluções pode ser organizado em uma base de dados transacional. Os bons resultados obtidos da aplicação de diferentes versões do GRASP convencional para o PEC em [6] demonstraram a eficiência deste tipo de metaheurística quando aplicada ao problema. Estes resultados motivaram a escolha desta técnica de otimização para o trabalho descrito nesta dissertação, além de possibilitar que o mesmo fosse focado na comprovação da eficiência da introdução de memória na metaheurística através da incorporação de técnicas de MD e não na eficiência da própria técnica perante

métodos exatos.

As idéias e parte dos resultados apresentados nesta dissertação foram publicados em [30, 31, 32]. A análise dos resultados realizada no capítulo anterior evidenciou que o tipo de hibridação proposta trouxe melhorias à técnica convencional. Também ficou evidenciado que, dentre as quatro estratégias propostas, a de melhor desempenho global foi a que utilizou padrões maximais, denominada nMM. As técnicas híbridas, em geral, apresentaram melhorias sobre o algoritmo GRASP em termos de valores médios obtidos, de melhores valores encontrados e em desvios padrões nas soluções, o que demonstra estabilidade e robustez das propostas. A proposta nMM ainda foi confrontada com o GRASP convencional no intuito de demonstrar que a hibridação pode acelerar o processo de encontrar boas soluções.

Foi feito um experimento para investigar o impacto do tamanho do conjunto elite na qualidade das soluções obtidas pelo GRASP-MD. O teste, que reduz o tamanho da base de soluções a ser minerada, não trouxe melhoria significativa à técnica. No entanto, trabalhos futuros podem investigar o impacto do tamanho do conjunto elite através de testes feitos com diferentes valores para este parâmetro.

O exame a respeito das características do conjunto de padrões sugere que um fator de grande influência na qualidade do tipo de hibridação proposta é o número de padrões utilizados. Outro fator importante se refere à utilização de padrões maximais, o que reduz o grau de semelhança dos mesmos. Foi observado ainda que o suporte médio dos padrões utilizados tem maior influência na qualidade média das soluções que o tamanho médio dos mesmos, que aparenta ser o fator de menor peso na qualidade média das soluções, entre aqueles estudados. A técnica que melhor combinou as características dos padrões estudadas foi a técnica nMM, que obteve o melhor desempenho entre todas. As análises realizadas sugerem que as diferentes características têm influência diferenciada sobre o desempenho das estratégias. Desta forma, trabalhos futuros podem ser realizados no sentido de se analisar com mais detalhe e rigor a importância de cada uma delas.

Neste trabalho, o processo de Mineração de Dados é executado uma única vez. Propostas futuras podem alternar iterações GRASP com o procedimento de MD

mais vezes ao longo do processo, permitindo que a extração de padrões ocorra a partir de conjuntos elite cada vez mais refinados. Paralelizações cooperativas do GRASP-MD poderiam implementar sucessivos processos de MD sobre as soluções geradas por diversos processadores executando iterações híbridas, alimentando-os novamente com padrões obtidos sobre bases de soluções continuamente refinadas.

Em um âmbito mais amplo, diversas outras investigações poderiam ser propostas para hibridação de metaheurísticas com técnicas de Mineração de Dados, como introduzir estas mesmas técnicas de MD ao GRASP aplicado a outros problemas [32], introduzir outras técnicas de MD ao GRASP ou ainda avaliar a introdução de MD em diferentes metaheurísticas.

Outra possibilidade de trabalho futuro seria estabelecer uma comparação entre o trabalho desenvolvido nesta dissertação com trabalhos recentes [3, 8] que propõem idéias similares. Estes trabalhos abordam estratégias similares ao GRASP e também utilizam-se de informações contidas em soluções de elite para a obtenção de soluções melhores. No entanto, ao invés de utilizarem Mineração de Dados para extrair informação a partir do conjunto de soluções de elite, estes trabalhos utilizam a técnica de Construção de Vocabulário [12] para identificar blocos de informação comuns a soluções de elite e combiná-los. Desta forma, visam combinar estes blocos de informação, enquanto o trabalho desta dissertação utiliza-se de apenas um padrão de cada vez, não estabelecendo nenhum tipo de combinação entre os mesmos. Estas diferenças podem ser exploradas e o impacto de cada abordagem ser avaliado em investigações futuras.

Referências Bibliográficas

[1] Agrawal, R., Imielinski, T. e Swami, A. Mining association rules between sets of items in large databases. Em *SIGMOD '93: Proceedings of the 1993 ACM SIGMOD international conference on Management of data* (New York, NY, USA, 1993), ACM Press, pp. 207–216.

[2] Agrawal, R. e Srikant, R. Fast algorithms for mining association rules in large databases. Em *VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases* (San Francisco, CA, USA, 1994), Morgan Kaufmann Publishers Inc., pp. 487–499.

[3] Aloise, D. Heurísticas para o projeto de redes com funções de custo discretas. Tese de Mestrado, Pontifícia Universidade Católica do Rio de Janeiro, 2005.

[4] Bastide, Y., Taouil, R., Pasquier, N., Stumme, G. e Lakhal, L. Mining frequent patterns with counting inference. *SIGKDD Explor. Newsl. 2*, 2 (2000), 66–75.

[5] Berry, M. J. e Linoff, G. *Data Mining Techniques: For Marketing, Sales, and Customer Support.* John Wiley & Sons, Inc., New York, NY, USA, 1997.

[6] Delorme, X., Gandibleux, X. e Rodriguez, J. GRASP for set packing problems. *European Journal of Operational Research 153* (2003), 564–580.

[7] Feo, T. A. e Resende, M. G. C. Greedy randomized adaptive search procedures. *Journal of Global Optimization 6* (1995), 109–133.

[8] Fernandes, E. L. R. Heurísticas para o problema de seqüenciamento de dna por hibridação. Tese de Mestrado, Pontifícia Universidade Católica do Rio de Janeiro, 2005.

[9] Gandibleux, X., Delorme, X. e T'Kindt, V. An ant colony optimization algorithm for the set packing problem. Em *ANTS Workshop* (2004), vol. 3172 of *Lecture Notes in Computer Science*, Springer, pp. 49–60.

[10] Garey, M. R. e Johnson, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. V. H. Freeman and Company, 1979.

[11] Glover, F. *Scatter search and path relinking*. McGraw-Hill Ltd., UK, Maidenhead, UK, England, 1999.

[12] Glover, F. e Laguna, F. *Tabu Search*. Kluwer Academic Publishers, Norwell, MA, USA, 1997.

[13] Goethals, B. e Zaki, M. J. Advances in frequent itemset mining implementations: Introduction to fimi03. Em *FIMI* [14].

[14] Goethals, B. e Zaki, M. J., Eds. *FIMI '03, Frequent Itemset Mining Implementations, Proceedings of the ICDM 2003 Workshop on Frequent Itemset Mining Implementations, 19 December 2003, Melbourne, Florida, USA* (2003), vol. 90 of *CEUR Workshop Proceedings*, CEUR-WS.org, disponível em http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS//Vol-90/, acessada em 20/09/2005.

[15] Grahne, G. e Zhu, J. Efficiently using prefix-trees in mining frequent itemsets. Em Goethals and Zaki [14].

[16] Guo, Y., Lim, A., Rodrigues, B. e Zhu, Y. Heuristics for a brokering set packing problem. Em *AI&M 1-2004, Eighth International Symposium on Artificial Intelligence and Mathematics* (Fort Lauderdale, Florida, USA, 2004), disponível em http://rutcor.rutgers.edu/∼amai/aimath04/AcceptedPapers/Guo-aimath04.pdf, acessada em 20/09/2005.

[17] Han, J. e Kamber, M. *Data mining: concepts and techniques.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000.

[18] Han, J., Pei, J. e Yin, Y. Mining frequent patterns without candidate generation. Em *SIGMOD '00: Proceedings of the 2000 ACM SIGMOD international conference on Management of data* (New York, NY, USA, 2000), ACM Press, pp. 1–12.

[19] Lodi, A., Allemand, K. e Liebling, T. An evolutionary heuristic for quadratic 0-1 programming. *European Journal of Operational Research 119* (1999), 662–670.

[20] Mingozzi, A., Maniezzo, V., Ricciardelli, S. e Bianco, L. An exact algorithm for the resource-constrained project scheduling problem based on a new mathematical formulation. *Management Science 44*, 5 (1998), 714–729.

[21] Mladenovic, N. e Hansen, P. *A tutorial on variable neighborhood search.* Technical Report G-2003-46, Les Cahiers du GERAD, HEC Montréal and GERAD, Canada, 2003.

[22] Nemhauser, G. L. e Wolsey, L. A. *Integer and combinatorial optimization.* Wiley-Interscience, New York, NY, USA, 1988.

[23] Orlando, S., Lucchese, C., Palmerini, P., Perego, R. e Silvestri, F. kdci: a multi-strategy algorithm for mining frequent sets. Em Goethals and Zaki [14].

[24] Orlando, S., Palmerini, P. e Perego, R. Enhancing the apriori algorithm for frequent set counting. Em *DaWaK '01: Proceedings of the Third International Conference on Data Warehousing and Knowledge Discovery* (London, UK, 2001), Springer-Verlag, pp. 71–82.

[25] Orlando, S., Palmerini, P., Perego, R. e Silvestri, F. Adaptive and resource-aware mining of frequent sets. Em *ICDM '02: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)* (Washington, DC, USA, 2002), IEEE Computer Society, pp. 338–345.

[26] Padberg, M. W. On the facial structure of set packing polyhedra. *Mathematical Programming 5* (1973), 199–215.

[27] Pardalos, P. M. e Rodgers, G. P. Computational aspects of a branch and bound algorithm for quadratic zero-one programming. *Computing 45*, 2 (1990), 131–144.

[28] Park, J. S., Chen, M.-S. e Yu, P. S. An effective hash-based algorithm for mining association rules. Em *SIGMOD '95: Proceedings of the 1995 ACM SIGMOD international conference on Management of data* (New York, NY, USA, 1995), ACM Press, pp. 175–186.

[29] Pietracaprina, A. e Zandolin, D. Mining frequent itemsets using patricia tries. Em Goethals and Zaki [14].

[30] Ribeiro, M. H., Plastino, A. e Martins, S. Hybridization of GRASP metaheuristic with data mining techniques. *Journal of Mathemathical Modelling and Algorithms* (2005), a ser publicado.

[31] Ribeiro, M. H., Trindade, V., Plastino, A. e Martins, S. Hybridization of GRASP metaheuristic with data mining techniques. *Workshop on Hybrid Metaheuristcs in conjunction with the 16th European Conf. on Artificial Intelligence* (2004), 69–78.

[32] Santos, L. F., Ribeiro, M. H., Plastino, A. e Martins, S. A hybrid GRASP with data mining for the maximum diversity problem. *Second International Workshop on Hybrid Metaheuristics, HM2005, Barcelona, Spain, August 29-30, 2005, Proceedings* (2005), 116–127.

[33] Savasere, A., Omiecinski, E. e Navathe, S. B. An efficient algorithm for mining association rules in large databases. Em *VLDB '95: Proceedings of the 21th International Conference on Very Large Data Bases* (San Francisco, CA, USA, 1995), Morgan Kaufmann Publishers Inc., pp. 432–444.

[34] Talbi, E.-G. A taxonomy of hybrid metaheuristics. *Journal of Heuristics 8*, 5 (2002), 541–564.

[35] Zwaneveld, P. J., Kroon, L. G., Romeijin, H. E., Salomon, M., Dauzères-Pérès, S., Hoesel, S. P. V. e Ambergen., H. W. Routing trains through railway stations: Model formulation and algorithms. *Transportation Science 30*, 3 (1996), 181–194.