Incorporando Mineração de Dados a uma Heurística GRASP/VND para o Problema do Caixeiro Viajante com Coleta e Entrega Envolvendo Único Tipo de Produto

Marcos Guerine, Isabel Rosseti, Alexandre Plastino
Instituto de Computação - Universidade Federal Fluminense (UFF)
Rua Passo da Pátria, 156 - Bloco E - CEP 24210-240 - Niterói/RJ - Brasil
{mguerine,rosseti,plastino}@ic.uff.br

RESUMO

A incorporação de técnicas de mineração de dados em metaheurísticas tem se mostrado uma combinação eficiente para diversos problemas de otimização. Neste trabalho, essa abordagem é explorada em uma heurística já desenvolvida, baseada em conceitos da metaheurística GRASP e da estratégia de busca local VND, para o problema do caixeiro viajante com coleta e entrega envolvendo um único tipo de produto. Esse problema difere daqueles para os quais a mineração já foi utilizada pois sua solução é definida pela ordem dos clientes e não por um subconjunto desses elementos. Essa diferença caracteriza a principal contribuição deste trabalho. Experimentos computacionais mostraram que a hibridização com processos mineração de dados beneficiou o algoritmo original, encontrando soluções melhores em um menor tempo computacional.

PALAVRAS CHAVE. Metaheurística Híbrida, 1-PDTSP, Mineração de Dados, Área de classificação principal (Metaheurística).

ABSTRACT

The incorporation of data mining techniques into metaheuristics has been an efficient combination for several optimization problems. In this work, this strategy is explored in a existing heuristic, based on the metaheuristic GRASP and the local improvement strategy VND, for the one-commodity traveling salesman problem with pickup and delivery. This problem differs from others where the data mining module was applied in the sense that its solution is defined by the order of all clients, not by a subset of these elements. This difference represents the main contribution of this work. Computational experiments showed that this hybridization with data mining process benefits the pure algorithm both in average quality of solutions and execution time.

KEYWORDS. Hybrid Metaheuristic, 1-PDTSP, Data Mining, Main area (Metaheuristic).

1. Introdução

Nas últimas décadas, algoritmos baseados em metaheurísticas ganharam notoriedade por apresentarem soluções de boa qualidade para diversos problemas de otimização combinatória em um tempo computacional aceitável. Cada metaheurística proposta na literatura é baseada em um paradigma distinto e oferece diferentes mecanismos que permitem escapar de ótimos locais, contrariamente às heurísticas construtivas ou de melhoria.

Mais recentemente, conceitos e processos de outras áreas de pesquisa foram testadas em conjunto com metaheurísticas. Técnicas de mineração de dados (MD) - que consistem na extração de conhecimento, na forma de regras e padrões, de uma base de dados de

maneira automática (Han e Kamber, 2006) - vêm sendo aplicadas com sucesso para aperfeiçoar heurísticas já consolidadas na literatura (Santos *et al*., 2008; Plastino *et al*., 2011). Regras de associação, padrões sequenciais, agrupamento de dados e mineração de conjuntos frequentes são exemplos de técnicas que extraem conhecimento de grandes bases de dados, que podem ser utilizadas em favor de heurísticas de formas diferentes.

A ideia principal da utilização de MD consiste em, a partir de um conjunto de soluções de boa qualidade, extrair partes dessas soluções que se repetem e usá-las para guiar a busca no espaço de soluções. Essas partes frequentes, chamadas de padrões, expressam características importantes das soluções.

Na literatura, a hibridização de metaheurísticas e MD foi inicialmente proposta por Ribeiro *et al*. (2004, 2006), que incorporaram a técnica de mineração de conjuntos frequentes à metaheurística *Greedy Randomized Adaptive Search Procedure* (GRASP) (Feo e Resende, 1995), obtendo resultados relevantes tanto em termos de qualidade de solução, quanto em termos de tempo computacional, quando aplicada ao problema do empacotamento de conjuntos. Essa abordagem, denominada pelo autores como DM-GRASP, também foi aplicada com êxito ao problema da diversidade máxima (Santos *et al*., 2005), ao problema da replicação de servidores *multicast* confiável (Santos *et al*., 2006), ao problema das p-medianas (Plastino *et al*., 2011) e mais recentemente ao problema de projeto de redes a 2-caminhos (2PNDP) (Barbalho *et al*., 2013).

Todos os trabalhos citados anteriormente possuem uma característica em comum: suas soluções são representadas por conjuntos de elementos e não levam em consideração a ordem desses elementos. No problema das p-medianas, por exemplo, a ordem das facilidades escolhidas não influencia no custo que será calculado. Entretanto, em alguns problemas de otimização combinatória, essa ordem tem papel importante, como é o caso do problema do caixeiro viajante com coleta e entrega envolvendo único tipo de produto (em inglês, *One-Commodity Pickup and Delivery Traveling Salesman Problem*, 1-PDTSP).

O 1-PDTSP foi proposto por Hernández-Pérez e Salazar-González (2004a) e consiste em uma generalização do Problema do Caixeiro Viajante (PCV) ao considerar que cada cliente possui uma determinada demanda pelo produto distribuído. No 1-PDTSP, o objetivo é encontrar o trajeto de menor custo que passe por todos os clientes, respeitando as restrições de demanda e de capacidade do veículo. Nesse problema, a ordem dos clientes é importante na definição da qualidade de solução. Em uma rota, por exemplo, a simples inversão de dois clientes pode influenciar no atendimento dos demais, provocando até a inviabilidade da solução.

Assim, este trabalho tem por objetivo inserir um módulo de mineração de dados em uma heurística já existente para o 1-PDTSP, baseada na metaheurística GRASP e na estratégia de busca local *Variable Neighborhood Descent* (VND), proposta por Hernández-Pérez *et al*. (2009). O 1-PDTSP é um problema que envolve ordem e, por isso, a representação da sua solução não é um subconjunto de elementos. Essa característica difere este trabalho de todos os outros em que a mineração foi aplicada e, portanto, representa sua principal contribuição. Os resultados computacionais mostraram que essa abordagem permite melhorar tanto a qualidade das soluções quanto o tempo computacional despendido para encontrar ótimos locais ou, até mesmo, ótimos globais.

O restante do artigo está organizado da seguinte forma. A Seção 2 apresenta o 1-PDTSP, bem como uma revisão bibliográfica. A Seção 3 descreve o GRASP/VND de

Hernández-Pérez *et al*. (2009) proposto para o 1-PDTSP e a Seção 4 apresenta como a técnica de mineração foi inserida nessa heurística. A Seção 5 compara os resultados computacionais obtidos com uma implementação do GRASP/VND de Hernández-Pérez *et al*. (2009). Finalmente, a Seção 6 apresenta as conclusões deste trabalho, juntamente com a proposta de alguns trabalhos futuros.

2. Descrição do problema e revisão bibliográfica

Considera-se um grafo completo $G(V, A)$, sendo V o conjunto de vértices representando os clientes e A o conjunto de arestas (i, j) que compõem as suas ligações. Dados um veículo com capacidade limitada Q e os valores das distâncias c_{ij} entre cada par (i, j) de clientes, e assumindo que cada cliente i tem uma demanda inteira q_i associada, o 1-PDTSP consiste em encontrar o circuito hamiltoniano de menor custo que atenda às demandas dos clientes, não excedendo a capacidade do veículo. O ponto em que o veículo inicia e termina sua rota é denominado depósito e, por se tratar de um único produto, a quantia coletada em um cliente pode ser usada para suprir a demanda de outro. Uma formulação matemática para o 1-PDTSP pode ser encontrada em (Hernández-Pérez e Salazar-González, 2004a).

Quando a capacidade do veículo é bastante grande, o 1-PDTSP coincide com o PCV e, portanto, é considerado $\mathcal{NP}$-Difícil. Ademais, examinar se existe uma solução viável para uma instância do 1-PDTSP é um problema $\mathcal{NP}$-Completo. Por outro lado, dada uma solução para o 1-PDTSP, a tarefa de verificar se essa rota é viável pode ser feita em $\mathcal{O}(n)$ (Hernández-Pérez e Salazar-González, 2004a).

Esse problema tem algumas aplicações reais. Por exemplo, pode ser utilizado no reposicionamento de estoques de uma empresa varejista, onde uma filial da empresa necessita de um produto que outra filial tenha em excesso. Pode ocorrer então um reposicionamento daquele produto, transferindo-o do local que esteja sobrando para o local onde há falta. Outro exemplo interessante é o de terminais de saque de dinheiro. Quando um terminal acusa falta de notas em estoque, o banco pode optar por reposicioná-las entre seus terminais, migrando daqueles que tem menor frequência de saques para aqueles de maior frequência (Hernández-Pérez e Salazar-González, 2004a).

Na literatura, existem algumas técnicas de solução para o 1-PDTSP. Em Hernández-Pérez e Salazar-González (2004a), foi apresentado um algoritmo *branch-and-cut* que era capaz de resolver instâncias de, no máximo, 60 clientes. Em seguida, os mesmos autores propuseram duas heurísticas (Hernández-Pérez e Salazar-González, 2004b) a fim de tratar instâncias maiores. A primeira consiste em uma heurística de busca local desenvolvida para fornecer limitantes superiores primais ao *branch-and-cut* anterior e a outra consiste em aplicar o algoritmo *branch-and-cut* considerando apenas um subconjunto de variáveis (associadas a arestas promissoras), reduzindo assim o espaço de busca.

Uma nova versão do algoritmo *branch-and-cut* foi proposta posteriormente em Hernández-Pérez e Salazar-González (2007), com a adição de um novo conjunto de restrições para o 1-PDTSP, baseadas em desigualdades válidas para o problema de roteamento de veículos capacitado. Martinovic *et al*. (2008) propuseram uma heurística baseada em *Simmulated Annealing* (SA), modificada e iterativa, que utiliza uma construção gulosa com aleatoriedade.

Hernández-Pérez *et al*. (2009) propuseram uma heurística híbrida com componentes da metaheurística GRASP (Feo e Resende, 1995) e da estratégia de busca local *Va-*

riable Neighborhood Descent (VND) (Mladenović e Hansen, 1997). A solução inicial é construída iterativamente, selecionando um novo cliente a partir de uma lista restrita de candidatos, que é ordenada por um critério guloso e adaptativo atualizado a cada iteração. A fase de busca local do GRASP é então substituída pela heurística VND, que contém procedimentos de aprimoramento baseados nas trocas 2-opt e 3-opt, já conhecidas em problemas de roteamento de veículos. A Seção 3 apresenta detalhadamente essa heurística.

Em Zhao *et al*. (2009), foi desenvolvido um Algoritmo Genético (AG) que possuía uma nova heurística construtiva para criar a população inicial, além de uma heurística de busca local para que o algoritmo convergisse mais rapidamente. Em Paes *et al*. (2010) foi proposto um algoritmo *multistart* baseado nas metaheurísticas GRASP e *Iterated Local Search*, além de usar a heurística VND com ordem aleatória de vizinhanças como método de busca local.

Recentemente, Mladenović *et al*. (2012) propuseram um algoritmo baseado na metaheurística *Variable Neighboorhood Search* (VNS) (Mladenović e Hansen, 1997) que usa uma nova e eficiente forma de verificar a viabilidade das soluções. Essa verificação é baseada em uma árvore binária indexada que armazena informações específicas das soluções e reduz bastante o esforço computacional necessário nas heurísticas de busca local.

Na próxima seção, será revisada a heurística híbrida proposta por Hernández-Pérez *et al*. (2009). Essa heurística foi escolhida como base da proposta híbrida do presente trabalho por ser uma estratégia competitiva para o 1-PDTSP e pelos resultados anteriores bem sucedidos de se incluir a técnica de mineração de dados em heurísticas do tipo GRASP.

3. Heurística híbrida GRASP/VND

A heurística híbrida apresentada em Hernández-Pérez *et al*. (2009) possui estrutura semelhante a da metaheurística GRASP clássica como mostra o Algoritmo 1. A estratégia é composta por um laço principal externo cujo critério de parada é o número máximo de iterações. Dentro do laço principal, ocorrem as fases de construção (linha 4) e de busca local (linha 5). Após o fim desse laço, uma nova busca local é aplicada à melhor solução encontrada (etapa chamada de "otimização final") com o intuito de melhorá-la ainda mais (linha 10).

Algoritmo 1 Heurística Híbrida para o 1-PDTSP

```
1:  GRASP/VND ( maxIter )
2:  f(x*) ← ∞;
3:  Para iter = 1 até maxIter faça
4:      x ← ConstruçãoGrasp();
5:      x ← VND_1(x);
6:      Se x é viável e f(x) < f(x*) então
7:          x* ← x ;
8:      Fim-se;
9:  Fim-para;
10: x* ← VND_2(x*);
11: Retorne x*;
```

Na fase de construção, um cliente é inicialmente selecionado de maneira aleatória para representar o depósito. A seguir, novos clientes são selecionados e inseridos iterativamente na solução da seguinte forma: para cada iteração, os clientes que não inviabilizam a

solução são ordenados de maneira crescente de acordo com a sua distância para o último cliente inserido, e os l primeiros são incluídos na Lista Restrita de Candidatos (LRC). Se nenhum cliente puder ser inserido sem inviabilizar a solução, a LRC é então composta pelos l clientes mais próximos do último inserido. No algoritmo, o parâmetro l assume o valor de 10. Por fim, um cliente é escolhido aleatoriamente da LRC e inserido na solução em construção. Esta etapa termina quando todos os clientes são inseridos.

A busca local interna, denominada VND_1, é baseada na heurística VND, que consiste em aplicar múltiplas estruturas de vizinhança em ordem predefinida à solução de entrada, e sempre que houver melhoria na solução corrente, deve-se retornar à primeira vizinhança. O procedimento VND_1 é composto por variações das heurísticas clássicas de troca 2-opt e 3-opt, com pequenas modificações para aceitarem soluções inviáveis de entrada, e aplicados na ordem crescente de esforço computacional. Primeiro a heurística 2-opt, que consiste em remover duas arestas não adjacentes e adicionar duas novas a fim de formar uma nova rota. Em seguida, aplica-se a heurística 3-opt, que segue a mesma lógica que a heurística anterior, porém três arestas são removidas.

Após o fim do laço principal, a "otimização final" acontece com outra heurística VND, denominada VND_2, que é aplicada à melhor solução encontrada até o momento. O procedimento VND_2 é composto pela divisão da estrutura de vizinhança *Insertion*, também usada em problemas de roteamento de veículos, em dois operadores, aplicados em uma dada ordem: o primeiro, *Insertion Forward*, consiste em remover um cliente e reinseri-lo em uma posição posterior à que foi retirado. O segundo, *Insertion Backward*, bem similar ao primeiro, porém o cliente removido é reinserido em posições anteriores.

4. Incorporando Mineração de Dados: Heurística DM-GRASP/VND

Na área de Mineração de Dados, existem diversas técnicas de extração de regras e padrões de base de dados. Dentre elas, está a técnica de mineração de conjuntos frequentes (MCF), que consiste em extrair padrões de uma base de dados de transações, onde cada transação é formada por um conjunto de elementos do domínio de aplicação.

Neste trabalho, cada transação é, na verdade, uma solução para o 1-PDTSP e a base de dados consiste em um conjunto de soluções de boa qualidade, denominado conjunto elite (CE). O CE é construído e atualizado durante as primeiras n iterações do algoritmo original e armazena as d melhores soluções distintas encontradas. Por esse motivo, o CE pode ser entendido como uma forma de adicionar memória ao GRASP.

Após a construção do CE, todos os pares de clientes consecutivos (i, j) do CE são mapeados em identificadores únicos, a fim de viabilizar o processo de mineração, preservando a ordem dos clientes. Em seguida, aplica-se o minerador e os p padrões extraídos são remapeados na forma de arestas (i, j). Para o restante do algoritmo, a construção original é substituída por uma construção adaptada, que faz o uso desses padrões como base para construir novas soluções em mais n iterações.

Cada padrão minerado é formado por um conjunto de arestas que se repetiram juntas em sup_{min} soluções do CE, parâmetro conhecido como suporte mínimo. A quantidade e o tamanho dos padrões minerados variam conforme este valor. Dentro do padrão, uma aresta (i, j) possui um cliente de origem i e um cliente de destino j. Por ser frequente em sup_{min} soluções distintas, a aresta (i, j) indica que os clientes i e j foram visitados consecutivamente em sup_{min} soluções. Além disso, pode ocorrer de, no mesmo padrão, duas

(ou mais) arestas serem sequenciais e, se conectadas, geram segmentos de rota maiores que são denominados de componentes conexas (CC). Dessa forma, cada padrão é formado por uma ou mais CCs, identificadas e ordenadas de maneira decrescente de tamanho.

O Algoritmo 2 apresenta a fase de construção adaptada. Em cada construção, um dos p padrões é selecionado de maneira *round-robin* (linha 2). Após isso, uma CC é escolhida de acordo com a quantidade de vezes que aquele padrão foi utilizado. Na primeira vez, escolhe-se a maior CC, na segunda vez a segunda maior e assim por diante (linha 3).

Algoritmo 2 Construção adaptada para utilizar padrões

1: ConstruçãoAdaptada(*listaPadrões*, *conjuntoEliteSoluções*)
2: $p \leftarrow$ SelecionaPadrão(*listaPadrões*);
3: $cc \leftarrow$ SelecionaCC(p);
4: $s_{esc} \leftarrow$ SelecionaSoluçãoQueContémCC(cc,*conjuntoEliteSoluções*);
5: $s \leftarrow$ ExtraiSubrota(cc,s_{esc});
6: $s \leftarrow$ ConstruçãoGrasp(s);
7: **Retorne** s;

Uma vez selecionada a CC, a construção de uma nova solução s é guiada da seguinte maneira: identificam-se todas as soluções do CE que contêm a CC em sua rota e escolhe-se uma aleatoriamente, s_{esc} (linha 4). A solução s recebe inicialmente uma parte da rota de s_{esc}, desde o depósito até o fim da CC em s_{esc} (linha 5). A partir desse ponto, a rota distinta é completada, inserindo os clientes sempre no fim da solução, aplicando a mesma ideia da heurística construtiva original (linha 6).

Algoritmo 3 Heurística Híbrida com Mineração de Dados

1: DM-GRASP/VND (*maxIter*, sup_{min})
2: $f(x^*) \leftarrow \infty$;
3: $CE \leftarrow \emptyset$;
4: **Para** $iter = 1$ até *maxIter/2* **faça**
5: $x \leftarrow$ ConstruçãoGrasp();
6: $x \leftarrow VND_1(x)$;
7: **Se** x é viável e $f(x) < f(x^*)$ **então**
8: $x^* \leftarrow x$;
9: **Fim-se**;
10: AtualizaConjuntoEliteDeSoluções(x,CE);
11: **Fim-para**;
12: *listaPadrões* $\leftarrow$ ExecutaMineração(CE,sup_{min});
13: **Para** $iter = 1$ até *maxIter/2* **faça**
14: $x \leftarrow$ ConstruçãoAdaptada(*listaPadrões*,CE);
15: $x \leftarrow VND_1(x)$;
16: **Se** x é viável e $f(x) < f(x^*)$ **então**
17: $x^* \leftarrow x$;
18: **Fim-se**;
19: **Fim-para**;
20: $x^* \leftarrow VND_2(x^*)$;
21: **Retorne** x^*;

O Algoritmo 3 mostra a heurística híbrida com mineração de dados e as modificações em relação ao Algoritmo 1, representadas nas linhas 10,12 e 14. É possível observar que o Algoritmo 3 é composto por duas etapas idênticas ao Algoritmo 1, separando-se metade das iterações para cada etapa. O CE é construído na primeira etapa (linha 10), o minerador é aplicado entre as duas etapas (linha 12) e a nova construção é inserida na segunda etapa, substituindo a construção original (linha 14). Esse algoritmo foi denominado DM-GRASP/VND.

5. Resultados Computacionais

A heurística GRASP/VND de Hernández-Pérez *et al*. (2009) foi implementada na linguagem C++, bem como a heurística DM-GRASP/VND proposta neste trabalho. Testes computacionais foram executados em um computador equipado com processador Intel®Core^{TM} i5 CPU 650 @ 3.20GHz, com 4GB de memória RAM e Sistema Operacional Linux Ubuntu versão 10.10. Foram consideradas as maiores instâncias do 1-PDTSP propostas em Hernández-Pérez *et al*. (2009).

O número de iterações *maxIter*, o tamanho do conjunto elite d, o valor de suporte mínimo sup_{min}, o número de padrões p e a quantidade de iterações reservadas n para construir o CE foram, respectivamente, 200, 10, 20%, 10 e *maxIter*/2. Com exceção do número de iterações, os valores dos demais parâmetros foram definidos a partir das recomendações encontradas em (Plastino *et al*., 2011).

A Tabela 1 apresenta os resultados obtidos por cada heurística após dez execuções para cada instância. A tabela reporta, para cada instância, a melhor solução obtida, o valor médio de solução relativo às dez execuções e o tempo computacional médio das heurísticas GRASP/VND e DM-GRASP/VND. Além disso, apresenta a diferença percentual (Dif %) da heurística DM-GRASP/VND em relação ao GRASP/VND de Hernández-Pérez *et al*. (2009), e o desvio padrão obtido. Para cada grupo de instâncias do mesmo tamanho calculou-se a média parcial das diferenças percentuais e, ao final da tabela, calculou-se a média geral. Na comparação entre os algoritmos, os valores em negrito representam os melhores resultados obtidos.

Observando a Tabela 1, é possível notar que, para todas as instâncias, a heurística DM-GRASP/VND apresenta as melhores soluções médias, em menores tempos computacionais. Apenas em cinco instâncias de um total de 50, o DM-GRASP/VND não conseguiu superar o GRASP/VND em relação a melhor solução obtida. O ganho percentual geral do DM-GRASP/VND foi em média igual a 1.34%, sendo, em média, 30.63% mais rápido em relação ao GRASP/VND original.

A redução do tempo está diretamente relacionada a dois fatores e pode ser verificada na Figura 4. Primeiro, a construção com padrões é mais rápida, pois usa parte de soluções do conjunto elite como base e, segundo, a qualidade da solução construída após a mineração que, por ser melhor que a solução obtida na construção original, garante que a busca local convirja mais rapidamente a um ótimo local.

Nas seções a seguir, são apresentados alguns experimentos adicionais a fim de ilustrar e comparar o comportamento dos dois algoritmos analisados neste trabalho.

Tabela 1. Resultados Computacionais do GRASP/VND e DM-GRASP/VND

Instância		GRASP/VND			DM-GRASP/VND							
		Melhor Solução	Media Solução	Tempo Médio	Melhor Solução	Dif % Melhor	Media Solução	Dif % Media	Desvio Padrão	Tempo Médio	Dif % Tempo	Desvio Padrão
n100q10	A	12369	12514.4	4.01	**11915**	-3.67	**12375.5**	-1.11	178.29	**2.97**	-25.98	0.14
n100q10	B	13668	13885.7	3.86	**13596**	-0.53	**13823.1**	-0.45	116.00	**2.77**	-28.07	0.07
n100q10	C	14619	14810.8	4.01	**14310**	-2.11	**14603**	-1.40	139.97	**2.85**	-28.92	0.12
n100q10	D	14806	14993.4	4.15	**14666**	-0.95	**14772.7**	-1.47	99.59	**3.12**	-24.76	0.08
n100q10	E	12594	12819.7	3.94	**12018**	-4.57	**12587.1**	-1.81	272.30	**2.63**	-33.27	0.09
n100q10	F	12082	12297.2	3.57	**11891**	-1.58	**12125.1**	-1.40	183.01	**2.67**	-25.24	0.11
n100q10	G	12344	12623.4	3.84	**12176**	-1.36	**12481.5**	-1.12	176.63	**2.71**	-29.56	0.12
n100q10	H	13405	13590.7	3.72	**13362**	-0.32	**13459.8**	-0.96	80.20	**2.68**	-27.93	0.08
n100q10	I	**14512**	14715.9	3.74	14514	0.01	**14698**	-0.12	171.33	**2.60**	-30.58	0.10
n100q10	J	**13700**	13992.0	4.00	13713	0.09	**13905.9**	-0.62	129.06	**2.99**	-25.28	0.09
Média Parcial						-1.50		-1.05			-27.96	
n200q10	A	18707	19053.1	34.34	**18319**	-2.07	**18725.7**	-1.72	234.91	**24.00**	-30.10	0.75
n200q10	B	19046	19406.7	33.27	**18689**	-1.87	**19273.4**	-0.69	314.77	**21.90**	-34.18	0.50
n200q10	C	17445	17740.2	37.19	**17430**	-0.09	**17630.7**	-0.62	127.86	**27.45**	-26.17	1.40
n200q10	D	22428	22772.4	33.65	**22047**	-1.70	**22524.4**	-1.09	327.11	**22.69**	-32.58	1.02
n200q10	E	20409	20738.2	36.77	**20323**	-0.42	**20639.7**	-0.47	204.88	**24.63**	-33.02	0.48
n200q10	F	22483	22709.4	37.10	**22295**	-0.84	**22615.9**	-0.41	199.50	**27.22**	-26.63	1.11
n200q10	G	18585	18855.3	34.72	**18147**	-2.36	**18735.5**	-0.64	265.92	**21.81**	-37.16	1.14
n200q10	H	22165	22588.2	39.85	**21907**	-1.16	**22348.4**	-1.06	217.83	**26.65**	-33.12	1.31
n200q10	I	19533	19859.3	34.22	**19362**	-0.88	**19504.1**	-1.79	75.28	**22.76**	-33.47	0.94
n200q10	J	20179	20471.6	32.80	**20011**	-0.83	**20244.1**	-1.11	184.81	**23.15**	-29.42	0.55
Média Parcial						-1.22		-0.96			-31.59	
n300q10	A	24942	25148.1	136.01	**24392**	-2.21	**24738.4**	-1.63	159.43	**92.17**	-32.23	2.91
n300q10	B	24413	24802.3	133.15	**24347**	-0.27	**24595**	-0.84	177.65	**89.63**	-32.68	3.61
n300q10	C	23212	23418.2	142.24	**22838**	-1.61	**23170.2**	-1.06	166.65	**92.90**	-34.69	2.26
n300q10	D	27080	27614.3	147.46	**26325**	-2.79	**27113.1**	-1.82	399.42	**99.18**	-32.74	2.29
n300q10	E	28643	28914.2	147.16	**27980**	-2.31	**28425.1**	-1.69	255.94	**99.90**	-32.11	3.69
n300q10	F	25843	26213.9	143.07	**25592**	-0.97	**25895.3**	-1.22	215.77	**108.49**	-24.17	4.38
n300q10	G	25631	25814.5	144.66	**25105**	-2.05	**25413.8**	-1.55	194.70	**108.70**	-24.86	2.88
n300q10	H	23590	23795.3	138.41	**23143**	-1.89	**23512.1**	-1.19	225.31	**93.02**	-32.79	2.22
n300q10	I	26018	26358.4	136.85	**25444**	-2.21	**25965.2**	-1.49	235.24	**94.40**	-31.02	2.75
n300q10	J	24050	24466.0	140.90	**23806**	-1.01	**24139.1**	-1.34	211.37	**98.85**	-29.84	2.75
Média Parcial						-1.73		-1.38			-30.71	
n400q10	A	33087	33266.8	393.04	**32170**	-2.77	**32620.1**	-1.94	190.60	**282.19**	-28.20	5.43
n400q10	B	26677	26797.2	347.47	**26107**	-2.14	**26395.1**	-1.50	189.54	**246.68**	-29.01	9.14
n400q10	C	30394	30682.2	399.14	**29838**	-1.83	**30235.7**	-1.46	301.90	**269.07**	-32.59	10.74
n400q10	D	25814	26267.5	400.79	**25291**	-2.03	**25750.1**	-1.97	324.48	**264.62**	-33.98	7.17
n400q10	E	26795	27313.9	355.53	**26393**	-1.50	**26824.5**	-1.79	251.92	**260.04**	-26.86	9.42
n400q10	F	**28107**	28910.0	361.85	28188	0.29	**28539.2**	-1.28	227.36	**256.23**	-29.19	12.21
n400q10	G	25697	26220.6	398.57	**25113**	-2.27	**25492.7**	-2.78	241.05	**279.50**	-29.88	11.70
n400q10	H	27158	27773.1	393.40	**26813**	-1.27	**27238.1**	-1.93	230.00	**278.53**	-29.20	8.46
n400q10	I	**30115**	30898.7	387.77	30208	0.31	**30549.5**	-1.13	209.11	**263.87**	-31.95	6.89
n400q10	J	27655	28059.0	383.00	**26921**	-2.65	**27536.1**	-1.86	300.73	**268.10**	-30.00	12.73
Média Parcial						-1.59		-1.76			-30.08	
n500q10	A	29874	30661.4	825.944	**29558**	-1.06	**30246.4**	-1.35	338.46	**579.69**	-29.82	21.64
n500q10	B	28559	29042.9	846.077	**28253**	-1.07	**28583.9**	-1.58	237.47	**573.82**	-32.18	20.44
n500q10	C	32360	33162.5	867.237	**32065**	-0.91	**32569.1**	-1.79	287.52	**577.93**	-33.36	25.49
n500q10	D	32750	33074.3	863.707	**32117**	-1.93	**32484.5**	-1.78	205.99	**593.99**	-31.23	32.04
n500q10	E	32298	32667.1	881.043	**31704**	-1.84	**32263.6**	-1.24	319.04	**598.28**	-32.09	33.37
n500q10	F	30856	31354.6	813.255	**30432**	-1.37	**30991.2**	-1.16	289.39	**511.36**	-37.12	14.57
n500q10	G	28879	29123.4	885.222	**28357**	-1.81	**28642.5**	-1.65	227.30	**597.69**	-32.48	32.32
n500q10	H	38579	39023.5	849.812	**37926**	-1.69	**38350.5**	-1.72	303.54	**596.57**	-29.80	21.47
n500q10	I	32718	33217.7	858.721	**32330**	-1.19	**32624.5**	-1.79	187.99	**547.15**	-36.28	20.10
n500q10	J	**32407**	33131.7	873.12	32530	0.38	**32720.7**	-1.24	170.27	**576.76**	-33.94	26.92
Média Parcial						-1.25		-1.53			-32.83	
Média Geral						**-1.46%**		**-1.34%**			**-30.63%**	

5.1. Significância estatística

Para verificar que os ganhos referentes à média de solução reportados na Tabela 1 têm significância estatística, foi realizado o teste t de *Student* pareado, com p-valor igual a 0.05.

A Tabela 2 mostra a comparação entre os algoritmos GRASP/VND e DM-GRASP/VND, separados por grupos de instâncias com mesmo número de clientes. O número fora dos parênteses indica em quantas instâncias aquela estratégia foi melhor em relação à média de solução, enquanto o valor entre parênteses indica o número de vezes em que o p-valor foi menor que 0.05, significando que a probabilidade de a diferença de desempenho dos algoritmos ser devido à aleatoriedade é menor que 5%.

É possível notar que quase todas as diferenças de desempenho entre os algoritmos

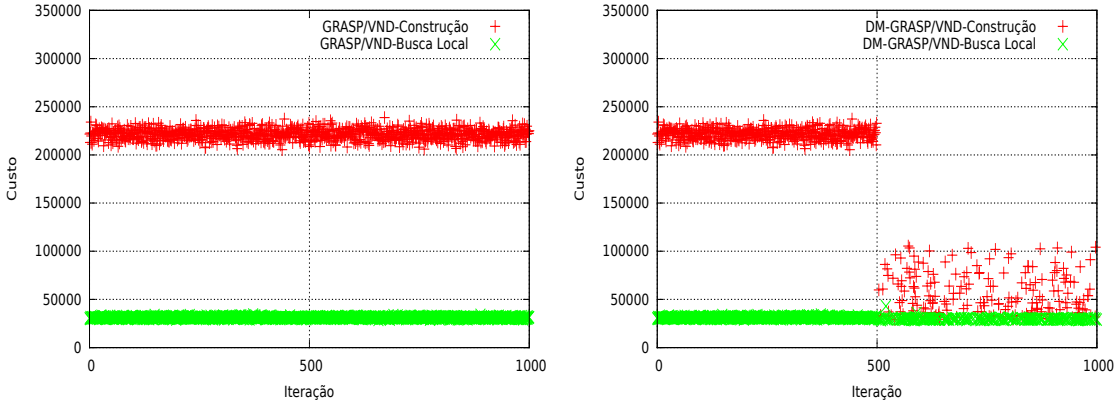
Tabela 2. Análise de significância estatística

Algoritmo	Tamanhos das instâncias				
	n100	n200	n300	n400	n500
GRASP/VND	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)
DM-GRASP/VND	10(5)	10(5)	10(10)	10(10)	10(10)

têm significância estatística, com exceção dos grupos com 100 e 200 clientes, em que foi possível obter significância estatística em metade das instâncias.

5.2. Análise do comportamento das estratégias

As Figuras 1(a) e 1(b) mostram como se comportam as fases de construção e busca local em ambos os algoritmos, reportando por iteração os valores de solução obtidos em cada uma das fases. Esse teste foi realizado para a instância n500q10G, executando 1000 iterações de cada estratégia.



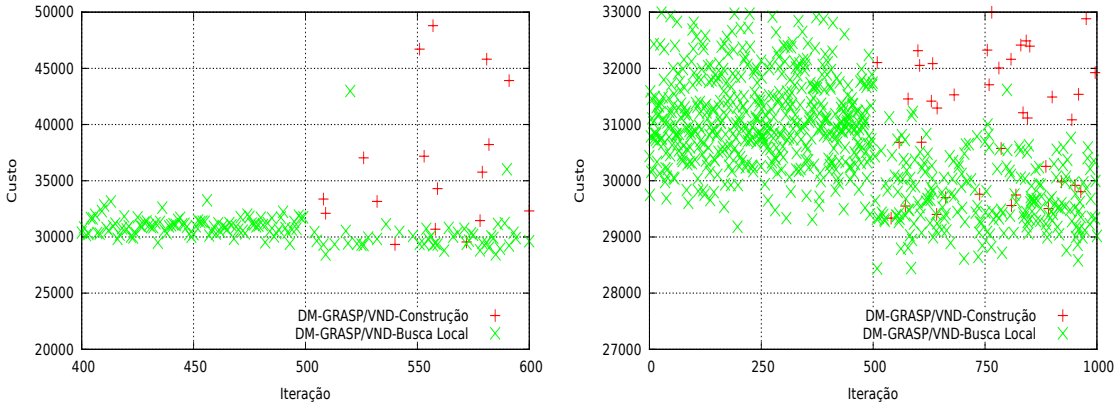
(a) GRASP/VND para a instância n500q10G (b) DM-GRASP/VND para a instância n500q10G

Figura 1. Gráfico Custo x Iteração

É possível notar que os dois algoritmos GRASP/VND e DM-GRASP/VND tem exatamente o mesmo comportamento até a iteração 500. No entanto, na iteração 500, a Figura 1(b) mostra o momento exato em que ocorre a mineração, revelando uma grande redução no custo das soluções construídas a partir de então. Já na Figura 1(a), a partir do mesmo ponto, a construção do GRASP/VND não sofre modificações.

Para que seja percebida a tendência de melhoria da busca local após a mineração, ampliou-se a Figura 1(b), como mostra a Figura 2. Na Figura 2(a), reduziu-se o intervalo de iterações, focando na vizinhança da iteração 500, onde ocorre a mineração. Pode-se notar que após a mineração, as soluções começam a atingir com maior frequência soluções de menor custo, com valores abaixo de 30000.

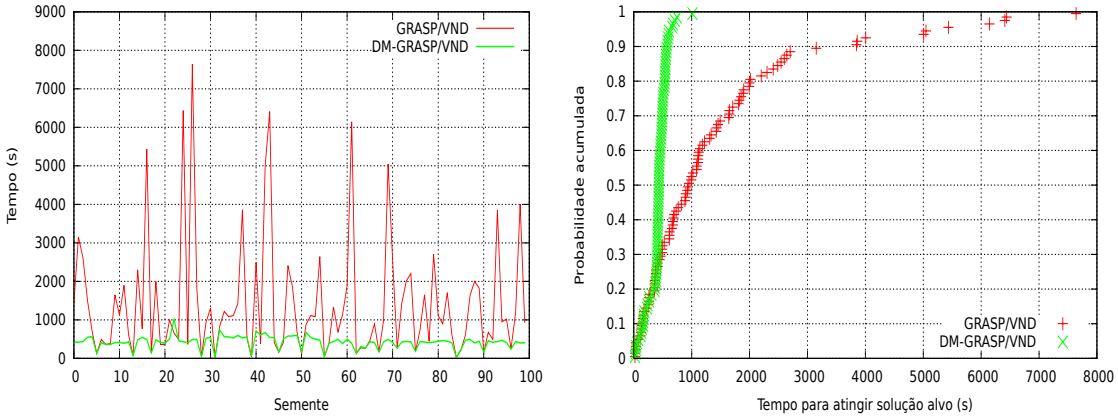
Na Figura 2(b), são apresentados os custos das soluções obtidas durante todas as 1000 iterações, porém ampliando o intervalo do eixo de custo entre os valores de 28000 e 32000. É possível perceber que antes da iteração 500, o algoritmo não encontrava soluções inferiores a 29000 e, após a iteração 500, esse valor foi alcançado em diversas iterações. Outro fator importante é que existem soluções construídas após a mineração que possuem custos de solução tão bons quanto as soluções já exploradas pela busca local.



(a) Soluções após a mineração (b) Mapa de soluções

Figura 2. DM-GRASP/VND - Custo x Iteração ampliado

Em outra análise, ambos os algoritmos foram executados com 100 sementes aleatórias e foram coletados os tempos necessários para encontrar uma solução alvo. A instância escolhida foi a n500q10G e o valor alvo adotado foi 29123. A Figura 3(a) mostra o tempo de execução de cada algoritmo para atingir a solução alvo em cada semente escolhida. Pode-se observar que o algoritmo DM-GRASP/VND se mostra bem mais eficiente, alcançando a solução alvo quase sempre antes que o GRASP/VND.



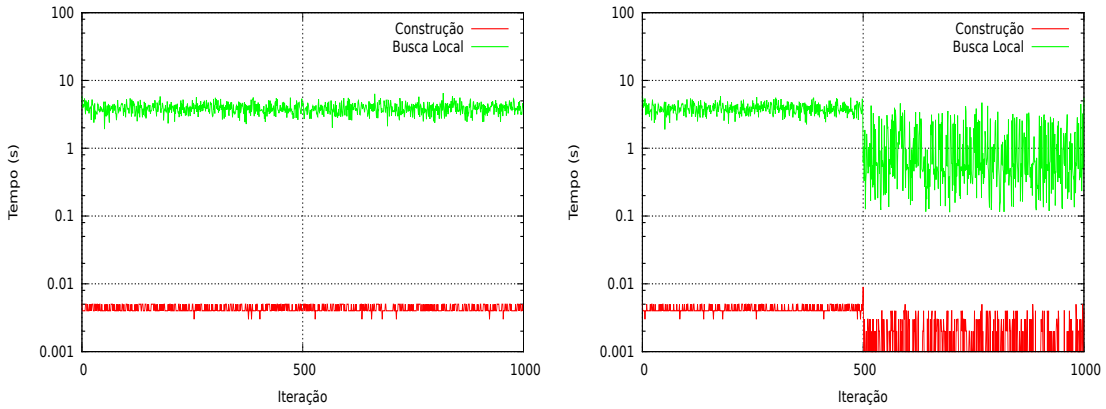
(a) Análise de convergência (b) Gráfico TTT-Plot

Figura 3. Análise com solução alvo para a instância n500q10G

A Figura 3(b) representa outra comparação entre os dois algoritmos abordados, baseada nos gráficos *Time-to-Target Plots* (TTT-plots) (Aiex *et al.*, 2006), que são usados para analisar o comportamento de algoritmos com componentes aleatórias. Esses gráficos mostram a probabilidade acumulada, no eixo das ordenadas, de um algoritmo encontrar uma solução melhor ou igual a uma solução alvo prefixada, em um tempo de execução definido no eixo das abscissas.

É possível notar que o comportamento da versão híbrida com mineração supera o GRASP/VND puro. A probabilidade do DM-GRASP/VND, por exemplo, encontrar a solução alvo em 1000 segundos é de quase 100% enquanto que para o GRASP/VND essa probabilidade está em torno de 55%.

Finalmente, para justificar a redução do tempo computacional obtido pelo DM-GRASP/VND, foi feita uma comparação apresentada na Figura 4. Essa análise exibe o tempo despendido pelo GRASP/VND na Figura 4(a) e pelo DM-GRASP/VND na Figura 4(b) em cada uma das etapas de construção e busca local. É possível observar que o DM-GRASP/VND obteve uma redução significativa de tempo após a iteração 500 (onde ocorre a mineração de dados), tanto na construção, quanto na busca local, ilustrando que a construção adaptada é mais rápida e também que a busca local converge mais rapidamente.



(a) GRASP/VND (b) DM-GRASP/VND

Figura 4. Análise de tempo para a instância n500q10G

6. Conclusões e Trabalhos Futuros

Neste trabalho, foi apresentada uma heurística híbrida que incorpora uma técnica de mineração de dados a uma heurística previamente proposta e de desempenho competitivo para o 1-PDTSP. Foram realizados experimentos computacionais em 50 instâncias da literatura, com o número de clientes variando entre 100 e 500.

Os resultados indicaram o benefício da introdução de mineração de dados, conseguindo soluções melhores em um tempo de execução consideravelmente menor. O ganho em relação às melhores soluções foi em média de 1.46% e em relação às médias de solução esse ganho foi de 1.34%. O DM-GRASP/VND apresentou também uma redução média de 30.63% em termos de tempo de execução. Essas avaliações demonstram o bom desempenho do método proposto. Além disso, experimentos complementares comprovaram a superioridade da versão híbrida com mineração, no que diz respeito à convergência do método. Destaca-se ainda o fato de a maioria dos resultados reportados ter significância estatística.

Como trabalhos futuros, pretende-se utilizar o minerador mais de uma vez durante a execução do algoritmo. Essa estratégia já se mostrou eficaz para o problema das p-medianas (Plastino *et al.*, 2011) e também para o 2PNDP (Barbalho *et al.*, 2013).

Referências

Aiex, R. M., Resende, M. G. C., e Ribeiro, C. C. (2006). TTT plots: A perl program to create time-to-target plots. *Optimization Letters*, 1:10–1007.

Barbalho, H., Rosseti, I., Martins, S. L., e Plastino, A. (2013). A hybrid data mining GRASP with path-relinking. *Computers & Operations Research*. Artigo aceito para publicação.

Feo, T. A. e Resende, M. G. C. (1995). Greedy Randomized Adaptive Search Procedures. *Journal of Global Optimization*, 6:109–133.

Han, J. e Kamber, M. (2006). *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 2^a edição.

Hernández-Pérez, H. e Salazar-González, J. (2004a). A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. *Discrete Applied Mathematics*, 145:453–459.

Hernández-Pérez, H. e Salazar-González, J. (2004b). Heuristics for the one commodity pickup-and-delivery traveling salesman problem. *Transportation Science*, 38:245–255.

Hernández-Pérez, H. e Salazar-González, J. (2007). The one-commodity pickup and delivery traveling salesman problem: Inequalities and algorithms. *Networks*, 50:258–272.

Hernández-Pérez, H., Salazar-González, J., e Rodríguez-Martín, I. (2009). A hybrid GRASP/VND heuristic for the one-commodity pickup-and-delivery traveling salesman problem. *Computers & Operations Research*, 36:1639–1645.

Martinovic, G., Aleksi, I., e Baumgartner, A. (2008). Single-Commodity Vehicle Routing Problem with Pickup and Delivery Service. *Mathematical Problems in Engineering*, 2008:1–18.

Mladenović, N. e Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100.

Mladenović, N., Urosevic, D., Hanafi, S., e Ilic, A. (2012). A general variable neighborhood search for the one-commodity pickup-and-delivery travelling salesman problem. *European Journal of Operational Research*, 220:270–285.

Paes, B. C., Subramanian, A., e Ochi, L. S. (2010). Uma heurística hibrida para o Problema do Caixeiro Viajante com Coleta e Entrega envolvendo um único tipo de produto. Em *Anais do XLII Simpósio Brasileiro de Pesquisa Operacional*, páginas 1513–1524.

Plastino, A., Fuchshuber, R., Martins, S. L., Freitas, A. A., e Salhi, S. (2011). A hybrid data mining metaheuristic for the p-median problem. *Statistical Analysis and Data Mining*, 4:313–335.

Ribeiro, M. H., de Aragão Trindade, V., Plastino, A., e Martins, S. L. (2004). Hybridization of GRASP Metaheuristics with Data Mining Techniques. Em *Hybrid Metaheuristics*, páginas 69–78.

Ribeiro, M. H., Plastino, A., e Martins, S. L. (2006). Hybridization of GRASP Metaheuristic with Data Mining Techniques. *Journal of Mathematical Modelling Algorithms*, 5:23–41.

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

Santos, L. F., Milagres, R., Albuquerque, C. V., Martins, S. L., e Plastino, A. (2006). A Hybrid GRASP with Data Mining for Efficient Server Replication for Reliable Multicast. Em *Global Telecommunications Conference, 2006. GLOBECOM '06*, páginas 1–6.

Santos, L. F., Ribeiro, M. H., Plastino, A., e Martins, S. L. (2005). A Hybrid GRASP with Data Mining for the Maximum Diversity Problem. Em Blesa, M. J., Blum, C., Roli, A., e Sampels, M., editores, *Hybrid Metaheuristics*, volume 3636 of *Lecture Notes in Computer Science*, páginas 116–127. Springer Berlin Heidelberg.

Zhao, F., Li, S., Sun, J., e Mei, D. (2009). Genetic algorithm for the one-commodity pickup-and-delivery traveling salesman problem. *Computers & Industrial Engineering*, 56:1642 – 1648.