EXPLORANDO A HIBRIDIZAÇÃO DA HEURÍSTICA ILS COM MINERAÇÃO DE DADOS[1]

Pedro Yusim, Rafael Braga Morett,Alexandre Plastino, Simone L. Martins
Universidade Federal Fluminense
Niterói – RJ – Brasil
pedroyusim@hotmail.com, rafael.morett@gmail.com, {plastino, simone}@ic.uff.br

RESUMO

A hibridização de metaheurísticas tem sido utilizada para melhorar o desempenho dessas tanto em relação à qualidade das soluções obtidas quanto ao tempo computacional. Versões híbridas da metaheurística GRASP, que incorporam técnicas de mineração de dados, já foram desenvolvidas e obtiveram excelentes resultados. O objetivo deste trabalho é investigar como a utilização de mineração de dados pode ser utilizada para melhorar o desempenho da metaheurística ILS que se comporta de maneira diferente do GRASP. Para este estudo, o problema de cobertura de conjuntos por pares foi utilizado.

PALAVRAS CHAVE. Metaheurística Híbrida, ILS, Mineração de Dados.

Áreas: Metaheurística, Otimização Combinatória, Mineração de Dados.

ABSTRACT

The hybridization of metaheuristics has been used to improve their performance both in terms of quality of solutions as the computational time. Hybrid versions of the metaheuristic GRASP, which incorporates data mining techniques, have already been developed obtaining promising results. The objective of this study is to investigate how the use of data mining can be used to improve the performance of the ILS metaheuristic that behaves differently than GRASP. For this study, the set cover with pairs problem was used.

KEYWORDS. Hybrid Metaheuristic, ILS, Data Mining.

Areas: Metaheuristics, Combinatorial Optimization, Data Mining.

[1] Trabalho parcialmente financiado pelo CNPq e FAPERJ.

1. Introdução

Heurísticas e metaheurísticas são técnicas desenvolvidas com o objetivo de se obter soluções aproximadas para problemas de otimização combinatória computacionalmente difíceis. Existe um grande número de metaheurísticas, dentre as quais se destacam: algoritmos genéticos, busca tabu, colônias de formigas, GRASP, *iterated local search* (ILS), *scatter search* e *simulated annealing* (Glover e Kochenberger (2003)).

Recentemente, tem se evidenciado que a utilização de metaheurísticas híbridas que combinam metaheurísticas com outras técnicas de otimização pode prover um comportamento mais eficiente e uma maior flexibilidade para problemas reais e de tamanho muito grande (Blum *et al*. (2011)).

Mineração de Dados (Han e Kamber (2011)) consiste em uma linha de pesquisa e aplicação em banco de dados que visa criar estratégias específicas para extrair informações importantes, ocultas e previamente desconhecidas a partir de bases de dados na forma de regras e padrões. Dentre os diferentes tipos de informação que podem ser minerados, destacam-se: as regras de associação, modelos de classificação, padrões sequenciais e agrupamentos (*clusters*) de dados.

A ideia de se integrar técnicas de mineração de dados com metaheurísticas surgiu com o objetivo de se extrair padrões de boas soluções encontradas pela metaheurística original e utilizá-los para guiar buscas posteriores no espaço de soluções. O objetivo em hibridizar metaheurísticas com mineração de dados é obter resultados de qualidade superior aos encontrados pela aplicação da metaheurística original em tempos computacionais iguais ou menores.

Versões híbridas da metaheurística GRASP (*Greedy Randomized Adaptative Search Procedure*) (Resende e Ribeiro (2003)) que incorporam técnicas de mineração de dados apresentaram bons resultados tanto em relação à qualidade de soluções quanto ao tempo computacional (Plastino *et al.* (2009), Plastino *et al.* (2011), Ribeiro *et al.* (2006), Santos *et al.* (2006), Santos *et al*. (2008)). No contexto da hibridização do GRASP, um algoritmo de mineração é utilizado para reconhecer padrões em soluções sub-ótimas, que são utilizados para guiar a busca por melhores soluções.

Diante dos bons resultados obtidos com a hibridização do GRASP com mineração de dados, resolveu-se investigar a utilização de mineração em conjunto com outras metaheurísticas. O objetivo deste trabalho é apresentar a investigação realizada para integrar mineração de dados com a metaheurística ILS (*Iterated Local Search*), que tem sido amplamente utilizada para resolver problemas de otimização combinatória (Lourenço *et al*. (2003)). Neste estudo, foi utilizado o problema de cobertura de conjuntos por pares (PCCP) (Hassin e Segev (2005)).

O restante deste artigo está organizado da seguinte forma. O PCCP e a heurística ILS desenvolvida para resolvê-lo são descritos na Seção 2. Na Seção 3, apresentam-se as soluções propostas para utilizar mineração de dados em conjunto com a heurística ILS. Na Seção 4, são analisados os resultados dos experimentos computacionais obtidos pelas soluções propostas. Finalmente, na Seção 5, são apresentadas as conclusões e direções para trabalhos futuros.

2. Heurística ILS para o problema de cobertura de conjuntos por pares.

O PCCP foi definido em (Hassin e Segev (2005)) como uma generalização do problema de cobertura de conjuntos, conhecidamente NP-difícil. Muitas aplicações para esse problema podem ser encontradas em biologia computacional, como a inferência de haplótipos, e na área de redes de computadores, como monitoramento de redes e serviço de páginas Web.

O PCCP pode ser definido com o auxílio de uma função de cobertura. Considere os conjuntos $U = \{u_1, \ldots, u_n\}$ e $A = \{a_1, \ldots, a_m\}$, onde U representa o conjunto dos elementos a serem cobertos e A o conjunto dos objetos que, aos pares, estão associados a subconjuntos de elementos de U. A função de cobertura $C : A \times A \rightarrow 2^U$ especifica, para cada par de objetos $\{a_i, a_j\} \subseteq A$, o subconjunto dos elementos de U que são cobertos por este par. Sendo c_i um custo não negativo

associado ao objeto $a_i \in A$, o PCCP tem por objetivo encontrar uma coleção de objetos $S \subseteq A$ de tal forma que : $\bigcup_{\{a_i,a_j\} \subseteq S} C(a_i, a_j) = U$, onde o custo dado por $c(S) = \sum_{a_i \in S} c_i$ seja minimizado.

A metaheurística ILS (Lourenço *et al.(*2003)) consiste na construção de uma solução inicial e um posterior processo iterativo no qual se aplica, em cada iteração, uma perturbação na solução e um aprimoramento na solução perturbada através de uma busca local como mostrado na Figura 2.1. Uma solução inicial é determinada por um algoritmo construtivo (linha 1) e tenta-se buscar, em sua vizinhança, uma solução de melhor qualidade na etapa seguinte (linha 2). Depois disto, enquanto o critério de parada não for satisfeito, a cada iteração, a solução corrente passa por uma perturbação (linha 4) e, em seguida, sua vizinhança é explorada por um algoritmo de busca local (linha 5). Caso o critério de aceitação seja satisfeito, a solução obtida após a etapa de busca local passa a ser a solução corrente (linha 6).

```
procedimento ILS()
1.      sol ← Construção();
2.      sol ← Busca Local(sol);
3.      repita
4.          sol´ ← Perturbação(sol);
5.          sol´ ← Busca Local(sol´);
6.          sol  ← Critério de Aceitação(sol,sol´);
7.      até  Critério de Parada();
8.      retorne sol;
```

Figura 2.1: Pseudocódigo da metaheurística ILS

Uma questão importante a ser definida na implementação de uma heurística ILS é o nível de perturbação que será utilizado. Se o nível for baixo, a solução corrente pode ser pouco modificada e a busca local aplicada logo após pode fazer com que se retorne a solução anterior, causando pouca exploração do espaço de busca.

A heurística ILS desenvolvida em (Gonçalves (2010)) para o PCCP obteve bom desempenho tanto em qualidade de solução quanto em tempo computacional. A seguir, são descritas as heurísticas desenvolvidas para a construção da solução e etapa de busca local, a estratégia de perturbação, o critério de parada e de aceitação utilizados.

2.1. Heurística Construtiva Add Drop (HAD)

O objetivo desta heurística é construir uma solução inicial. Uma solução viável S é inicialmente criada contendo todos os objetos de A. Esses objetos são ordenados de modo não decrescente de acordo com seu custo associado e inseridos em A'. O primeiro objeto do conjunto ordenado A' é selecionado e removido. Em seguida, verifica-se se a remoção deste objeto da solução corrente S fará com que algum elemento de U fique descoberto. Se a remoção não causar inviabilidade na solução, este objeto é removido de S, caso contrário, permanece. O algoritmo termina quando não houver mais candidatos no conjunto A'.

2.2. Heurísticas de Busca Local

Para a etapa de busca local foram desenvolvidas duas heurísticas denominadas Busca Local Drop Add (BLDA) e Busca Local Troca Objeto (BLTO), que são utilizadas uma após a outra.

O algoritmo BLDA recebe como parâmetro uma solução S, obtida pelo algoritmo construtivo HDA descrito anteriormente, uma constante $\gamma \in [0, 1]$ e um inteiro positivo *k* que indica o critério de parada do algoritmo. A primeira etapa do algoritmo consiste em retirar alguns dos objetos da solução corrente. Para isto, inicialmente são escolhidos aleatoriamente $\gamma \times |S|$ objetos da solução corrente S formando o conjunto Q. Estes elementos são removidos da solução corrente e ficam impedidos de serem removidos nas próximas *I* iterações inserido-os numa lista tabu *lt*. Após esta etapa, os elementos de Q são retirados do conjunto solução, tornando a mesma inviável, ou seja, deixando descobertos alguns dos elementos de U. Para viabilizar a solução, realiza-se a reconstrução considerando apenas os elementos descobertos em U e gera-se a solução

S″. Na reconstrução, determina-se o conjunto de objetos que deve ser incluído na solução de forma a cobrir estes elementos com um menor custo.

Caso a solução S″ tenha um custo inferior ao custo da melhor solução encontrada até a iteração atual, esta é atualizada de forma a armazenar sempre a melhor solução. Este processo se repete enquanto melhores soluções estiverem sendo encontradas, sendo interrompido quando o algoritmo realizar *k* iterações sem atualizar a melhor solução corrente.

O algoritmo BLTO é executado em seguida. Com objetivo de determinar a melhor solução na vizinhança da solução corrente, remove-se um objeto presente no conjunto solução de cada vez e a solução é reconstruída utilizando uma heurística sorteada entre as heurísticas: HMP (Heurística do Melhor Par) e HOMP (Heurística do Objeto com Maior Participação). Estas heurísticas consistem na realização de iterações até que seja obtida uma solução viável. Em cada iteração, elemento(s) é (são) escolhido(s) para serem inseridos na solução de acordo com estratégias diferentes.

Na HMP, inserem-se os elementos que pertençam a um par de objetos que cobre um elemento descoberto e que acarrete no menor acréscimo no custo da solução corrente. Na HOMP, o elemento a ser inserido é aquele cuja inserção causará a cobertura do maior número de elementos descobertos. Para ambas as estratégias, há uma penalização para evitar que o objeto removido seja reinserido na solução. Por se tratar de um procedimento que adota a estratégia *best-improving*, a cada iteração, toda a vizinhança é analisada e a melhor solução na vizinhança de solução corrente é armazenada. Ocorre então a atualização da melhor solução corrente, caso tenha sido possível encontrar uma solução na vizinhança com custo inferior ao custo da melhor solução corrente. Caso a atualização ocorra, a busca reiniciará na vizinhança da nova solução. O algoritmo termina quando toda a vizinhança é analisada e nenhuma solução melhor que a melhor solução corrente for encontrada.

2.3. Estratégias de perturbação, critério de parada e de aceitação.

Na estratégia de perturbação, a solução corrente tem uma porcentagem dos seus elementos removidos, e a solução é reconstruída utilizando-se as heurísticas HMP ou HMOP que são escolhidas aleatoriamente. Como critério de parada do algoritmo, pode-se utilizar: número fixo de iterações, tempo limite de execução e limite de iterações sem que a melhor solução alcançada pelo ILS seja atualizada. Em cada iteração, a solução corrente é atualizada pela solução gerada após a busca local caso seu custo seja melhor do que o custo da solução corrente.

3. Estratégias de hibridização da heurística ILS com mineração de dados

Processos de mineração de dados são caracterizados pela extração automática de informações úteis a partir de bases de dados, na forma de regras e padrões. A idéia que motivou a aplicação desta técnica em conjunto com metaheurísticas é a possibilidade de se encontrar padrões que representem características das melhores soluções geradas pelas metaheurísticas originais para auxiliá-las na busca por soluções mais próximas do valor ótimo em um menor tempo computacional. Alguns trabalhos já utilizaram a ideia de guiar heurísticas através da obtenção de características de boas soluções previamente geradas utilizando outras técnicas diferentes de mineração de dados e desenvolvidas de forma específica para cada problema (Fleurent e Glover (1999), Lin e Kernighan (1973), Lodi e Libling (1999)). A utilização de mineração de dados pode levar ao desenvolvimento de heurísticas híbridas mais eficientes e mais generalistas, ou seja, podem ser aplicadas a qualquer tipo de problema.

Versões híbridas do GRASP que incorporam um módulo de mineração de dados vêm obtendo bom desempenho tanto em relação à qualidade de solução quanto em termos de tempo computacional (Plastino *et al.* (2009), Plastino *et al.* (2011), Ribeiro *et al.* (2006), Santos *et al.* (2006), Santos *et al*. (2008)). Surgiu então a ideia de investigar se a integração de mineração de dados com outras metaheurísticas poderia também melhorar o seu desempenho.

A metaheurística ILS vem obtendo bons resultados para resolver problemas de otimização combinatória e por isso foi escolhida para ser investigada neste trabalho. A heurística ILS desenvolvida especificamente para o PCCP (Gonçalves (2010)) foi utilizada porque obteve bom desempenho e seu código e instâncias estavam disponíveis.

A ideia geral de utilização de mineração de dados com a heurística ILS foi realizar um número de iterações da heurística ILS original para formar um repositório de soluções de boa qualidade. Posteriormente, obtêm-se padrões deste repositório utilizando-se uma técnica de mineração de dados. Neste estudo, utilizou-se o algoritmo FPmax* (Grahne e Zhu (2003)) para extrair conjuntos frequentes maximais – que foi utilizado nas heurísticas GRASP híbridas.

Estes padrões podem ser utilizados em iterações posteriores do ILS de duas formas distintas, no procedimento de perturbação da solução ou procedimento de busca local. Por limitação de espaço, será explorado apenas o segundo caso, por ter apresentado os melhores resultados.

A estrutura da estratégia ILS com mineração de dados está apresentada na Figura 3.1. O critério de parada utilizado é a realização de um número *n* de iterações, sendo que durante as primeiras *n/2* iterações executa-se o algoritmo ILS original e cria-se um repositório de *M* soluções (linhas 4 a 10). Os padrões são gerados realizando a mineração neste repositório (linha 11). Em seguida, são realizadas *n/2* iterações nas quais os padrões são utilizados no procedimento de busca local.

```
procedimento ILS_MD()
1.      sol ← Construcão();
2.      sol ← Busca Local(sol);
3.      it ← 1;
4.      repita
5.          sol´ ← Perturbação(sol);
6.          sol´ ← Busca Local(sol´);
7.          sol  ← Criterio de Aceitação(sol,sol´);
8.          Insere Rep(sol,Rep);
9.          it ← it + 1;
10.     até (it=n/2);
11.     padrões ← Mineração (Rep)
12.     repita
13.         sol´ ← Perturbação(sol);
14.         sol´ ← Busca Local(padrões,sol´);
15.         sol  ← Criterio de Aceitacao(sol,sol´);
16.         it ← it + 1;
17.     até (it=n);
18.     retorne sol;
```

Figura 3.1: Pseudocódigo da metaheurística ILS_MD

Foram implementadas seis estratégias de utilização dos padrões na busca local. Novamente por questões de espaço, serão somente apresentadas três estratégias que tiveram desempenhos variados.

ILS-MD-A: Utilizam-se os padrões na busca local BLTO de forma circular, não se permitindo que os elementos que pertencem ao padrão sejam removidos da solução durante a busca local.

ILS-MD-B: Utilizam-se os padrões na busca local BLDA de forma circular, não se permitindo que os elementos que pertencem ao padrão sejam removidos da solução durante a busca local, inserindo os padrões na lista tabu no início da busca local (depois de algumas iterações, o elemento pertencente à lista tabu perde a sua validade e volta a poder ser removido da solução).

ILS-MD-A-B: Utilizam-se os padrões nas buscas locais BLDA e BLTO de forma circular, não se permitindo que os elementos que pertencem ao padrão sejam removidos da solução durante a busca local BLDA, inserindo os padrões na lista tabu no início da busca local (depois de algumas iterações o elemento pertencente à lista tabu perde a sua validade e volta a poder ser removido da solução) e não deixando que os elementos que pertencem ao padrão sejam removidos da solução

durante a busca local BLTO.

4. Resultados experimentais

Nesta seção, apresentam-se os resultados dos experimentos computacionais realizados com as estratégias desenvolvidas. Os algoritmos foram implementados em C++ e compilados com g++ (gcc) 4.7.2. Os testes foram realizados em um computador AMD Phenom x6 Black 1100T (3.3GHz) com 8GB de memória RAM, executando Linux (kernel 3.5.0-24).

As instâncias scp_p utilizadas foram criadas em (Gonçalves(2010)). Elas possuem entre 50 e 28160 elementos e número de objetos variando entre 192 e 11264. O valor do parâmetro p afeta o número de pares da instância. Quanto menor o valor de p, maior é a quantidade de pares nas instâncias. Utilizando-se p = 25%, o número médio de pares é de 238.810, e com p = 75% este valor é de 77.543. Cada grupo scp_p possui 24 instâncias.

Em todas as estratégias, o repositório de soluções a ser minerado contém 10 soluções e foram utilizados os 10 maiores padrões minerados. Utilizou-se um suporte igual a 9 para a mineração. O critério de parada utilizado para todas as estratégias foi a execução de 100 iterações e, para avaliar cada estratégia, realizaram-se 10 execuções com sementes aleatórias distintas.

Em cada tabela de resultados, apresentam-se nas três primeiras linhas os resultados obtidos para cada grupo de instância, mostrando a diferença percentual entre o valor obtido executando-se a heurística híbrida e o valor obtido utilizando-se a heurística ILS. Diferenças percentuais positivas denotam que a heurística híbrida obteve melhores resultados que a heurística ILS e negativas denotam piores valores. Nas linhas seguintes, apresentam-se o valor médio da diferença percentual e o número de vitórias, derrotas e empates de cada heurística híbrida.

Na Tabela 4.1, são apresentados os valores médios de solução e os tempos computacionais referentes às estratégias que utilizam os padrões na busca local.

Grupo	**A**	**A-B**	**B**	**Tempo A**	**Tempo A-B**	**Tempo B**
scp_25	0,00399	0,01000	0,09439	26,83649	18,08747	-10,5828
scp_50	-0,07265	-0,04584	0,03978	23,78025	15,71879	-14,9251
scp_75	-0,02772	0,07745	0,0312	26,74326	15,39849	-2,83423
Média	-0,03213	0,01387	0,05512	25,78667	16,40158	-9,44739
Vit./Der./Emp.	28/39/5	40/30/2	47/22/3	72/0/0	71/1/0	18/54/0

Tabela 4.1: Valores médios de solução e tempo das diferentes heurísticas híbridas

Utilizando-se os padrões na busca local, os resultados obtidos em relação à qualidade da solução foram semelhantes aos obtidos pela heurística ILS, mas obtiveram-se resultados diversos em relação ao tempo computacional. As heurísticas ILS-MD-A e ILS-MD-A-B obtiveram tempos computacionais significativamente menores que a heurística ILS, enquanto que a ILS-MD-B obteve tempos maiores.

A heurística ILS-MD-A obteve um ganho em tempo computacional de 25% e a diferença percentual do valor médio da qualidade da solução ficou somente 0,032% abaixo da solução da heurística ILS. A heurística ILS-MD-A-B obteve um ganho em tempo de 16% e a diferença percentual do valor médio da solução ficou 0,014% acima da solução da heurística ILS.

Em um experimento adicional, cada uma das heurísticas híbridas foi executada durante o mesmo tempo que a heurística ILS. Foi possível observar que executando todas no mesmo tempo, das 72 instâncias, a ILS-MD-A-B obteve 60 melhores resultados e a ILS-MD-A, 53 melhores resultados.

Em outro experimento, executaram-se 100 iterações para a instância scp41_25 e obteve-se em cada iteração o tempo computacional utilizado pela fase de perturbação, busca local BLTO e

BLDA. Nos gráficos da Figura 4.1, o eixo X representa o número da iteração e o eixo Y, o tempo de execução de cada fase.

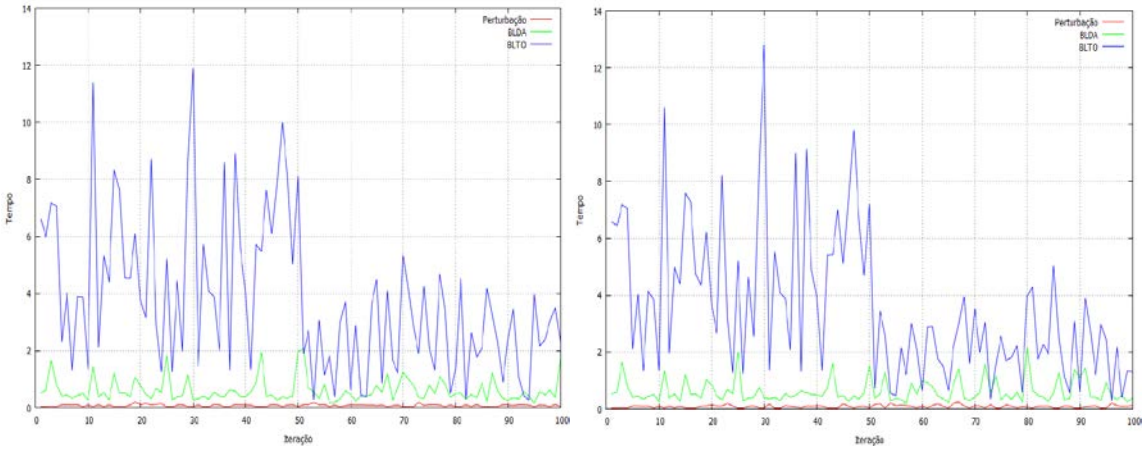
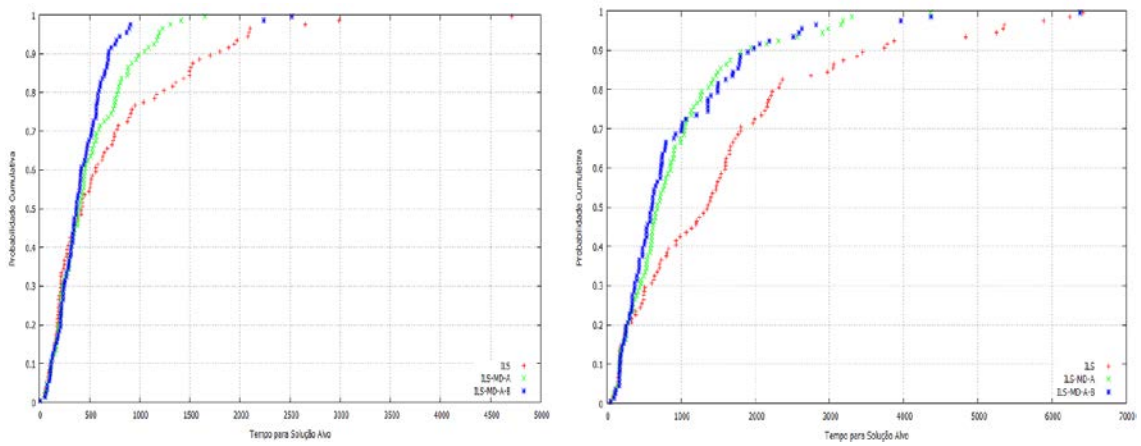


Figura 4.1: Tempo computacional utilizado pelas heurísticas ILS-MD-A e ILS-MD-A-B

Pode-se observar que na execução da heurística ILS-MD-A (primeiro gráfico), que utiliza os padrões na busca local BLTO, ocorre uma diminuição no tempo de execução da busca BLTO a partir da iteração 50 quando os padrões passam a ser utilizados. Na execução da heurística ILS-MD-A-B, quando os padrões são utilizados nas buscas BLTO e BLDA, pode-se observar o mesmo efeito. Estes experimentos indicam que a mineração de dados foi capaz de diminuir o tempo computacional para se obter soluções de mesma qualidade que a heurística ILS original.

Os gráficos da Figura 4.2 apresentam uma comparação entre as metaheurísticas ILS, ILS-MD-A e ILS-MD-A-B. Gráficos *time-to-target* (TTT) (Aiex *et al.* (2007)) podem ser utilizados para comparar o comportamento de heurísticas através dos seus tempos de execução. Um gráfico TTT é gerado executando-se um algoritmo várias vezes e medindo-se o tempo necessário para alcançar uma solução no mínimo tão boa quanto uma determinada solução alvo. Os gráficos da Figura 4.2 foram gerados através da execução de cada metaheurística 100 vezes, utilizando-se a instância scp41_25, com sementes aleatórias diferentes, para: (a) um alvo fácil e (b) um alvo difícil. Observa-se que tanto ILS-MD-A quanto ILS-MD-A-B alcançam os alvos mais rapidamente que ILS, comprovando que essas heurísticas híbridas demandam menor tempo computacional para atingir soluções de qualidade semelhante. A heurística ILS-MD-A-B atinge o alvo fácil mais rapidamente com uma probabilidade maior que a heurística ILS-MD-A, enquanto que ambas apresentam comportamento semelhante para alvos mais difíceis.



(a) (b)

Figura 4.2 : Gráficos *time-to-target* para a instância scp41_25 com alvo (a) fácil e (b) difícil

5. Conclusões

Neste artigo, foram propostas diversas heurísticas que hibridizam a estratégia ILS com mineração de dados. Algumas versões, que utilizaram padrões na busca local mostraram uma melhoria de desempenho em termos de tempo computacional, acelerando a execução e obtendo resultados de qualidade semelhante ou melhores que os resultados obtidos pela heurística ILS original.

Já haviam sido obtidos bons resultados com a hibridização de mineração de dados com a metaheurística GRASP. Os resultados apresentados neste artigo evidenciam que a utilização de mineração de dados em conjunto como a metaheurística ILS também pode trazer melhorias ao desempenho dessa importante metaheurística.

Referências

Aiex, R.M., Resende, M.G.C. e Ribeiro, C.C. (2007), TTTPLOTS: A perl program to create time-to-target plots, *Optimization Letters*, 1, 355-366.

Blum, C., Puchingerb, J., Raidl, G. R. e Roli, A. (2011), Hybrid metaheuristics in combinatorial optimization: A survey, *Applied Soft Computing*, 11, 4135–4151.

Festa, P. e Resende, M. (2002), GRASP: An annotated bibliography, em Ribeiro, C. C. e Hansen, P. (Eds.), *Essays and Surveys in Metaheuristics*, Kluwer Academic Publishers, 325-367.

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

Glover,F. e Kochenberger, **G.A.,** *Handbook of Metaheuristics*, *International Series in Operations Research & Management Science*; 57, Kluwer, 2003.

Gonçalves, L.B., Heurísticas para o Problema de Cobertura de Conjuntos por Pares, *Tese de Doutorado*, Universidade Federal Fluminense, 2010.

Han, J. e Kamber, **M.,** *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2011.

Hassin, R. e Segev, D. (2005), The set cover with pairs problem, *Lecture Notes in Computer Science*, 3821, Springer, 164–176.

Lin, S. e Kernighan, B.W. (1973), An effective heuristic algorithm for the traveling salesman problem.,*Operations Research*, 21:498-516.

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

Lourenço, H. R., Martin, O. C. e Stützle, T. (2003), Iterated local search, *Handbook of Metaheuristics*, 57, Springer - Kluwer Academic Publishers, New York, 320–353.

Plastino, A., Fonseca, E. R., Fuchshuber, R., Martins, S. L., Freitas, A. A., Luis, M., e Salhi, S. (2009), A hybrid data mining metaheuristic for the p-median problem, *SIAM International Conference on Data Mining*.

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

Resende, M. e Ribeiro, C., (2003), Greedy Randomized Adaptive Search Procedures, em Glover, F. e Kochenberger, G. (Eds.), *Handbook of Metaheuristics*, 219-249.

Ribeiro, M. H., Plastino, A. e Martins, S. L. (2006), Hybridization of GRASP Metaheuristic with Data Mining Techniques, *Journal of Mathematical Modeling and Algorithms*, 5:1, 23-41.

Santos, L. F. M., Milagres, R., Albuquerque, C., Plastino, A. e Martins, S. L. (2006), A Hybrid GRASP with Data Mining for Efficient Server Replication for Realiable Multicast, *Proceedings of the IEEE GLOBECOM Conference*.

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

Talbi, E. G. (2002), A Taxonomy of Hybrid Metaheuristics, *Journal of Heuristics*, 8:5, 541-564.