Uma Metaheurística Híbrida para o Problema de Recobrimento de Conjuntos com Agrupamento

Artur Ferreira Brum
Universidade Federal de Santa Maria - CT
Av. Roraima, 1000, Santa Maria, RS
arturfb@inf.ufsm.br

Olinto César Bassi de Araújo
Universidade Federal de Santa Maria - CTISM
Av. Roraima, 1000, Santa Maria, RS
olinto@ctism.ufsm.br

Felipe Martins Müller
Universidade Federal de Santa Maria - CT
Av. Roraima, 1000, Santa Maria, RS
felipe@inf.ufsm.br

RESUMO

Este trabalho aborda o problema de recobrimento de conjuntos com agrupamento, definido como uma variante do problema clássico de recobrimento de conjuntos, na qual os conjuntos estão dispostos em K agrupamentos disjuntos. São apresentadas duas contribuições para uma metaheurística baseada em Busca Tabu Iterada. A primeira abrange uma abordagem paralela para o componente de Busca Tabu, enquanto a segunda contribuição consiste na implementação de uma vizinhança baseada em programação inteira e que faz o uso de fixação de variáveis. A ideia subjacente ao uso da nova vizinhança é utilizar o tempo economizado na paralelização com o polimento de soluções promissoras obtidas pela Busca Tabu. Os experimentos computacionais mostram que o método proposto é capaz de encontrar 17 novas melhores soluções conhecidas para 30 instâncias do conjunto teste considerado.

PALAVRAS CHAVE. Otimização combinatória, Modelo matemático, Metaheurística Híbrida.

Área Principal: Otimização Combinatória, Metaheurísticas

ABSTRACT

This paper addresses the clustered set covering problem, defined as a variant of the classic set covering problem, in which sets are arranged in K disjoint clusters. Two contributions to an Iterated-Tabu-Search-based metaheuristic are presented. The former comprises a parallel approach for the Tabu Search component, while the latter contribution consists in the implementation of a neighborhood using integer programming and variable fixing. The idea behind the use of this neighborhood is to use the time saved by the parallelization with the polishment of promising solutions obtained by Tabu Search. The computational experiments show that the proposed method is able to find 17 new best-known solutions for 30 instances in a given test set.

KEYWORDS. Combinatorial optimization, Mathematical model, Hybrid Metaheuristic.

Main Area: Combinatorial Optimization, Metaheuristics

1. Introdução

Este trabalho aborda o problema de recobrimento de conjunto com agrupamento (PRCA), recentemente definido por Alfandari e Monnot (2014) . O PRCA consiste em uma variante NP-difícil do problema de recobrimento de conjuntos (PRC) clássico, que é amplamente conhecido e abordado em Ciência da Computação. Nessa variante os conjuntos de recobrimentos ficam dispostos em K agrupamentos disjuntos e sob cada um incide um custo fixo a ser pago uma única vez sempre que pelo menos um conjunto nele contido é selecionado.

O estudo do PRCA é motivado pela sua aplicabilidade em problemas práticos. Neste trabalho é demonstrado como o problema abordado por Bilal *et al.* (2014) , oriundo da indústria de mineração, pode ser modelado como PRCA. O problema envolve a avaliação de potenciais pontos de extração de minérios e inclui uma etapa em que o solo é analisado através de perfurações de caráter exploratório, chamadas sondagens. As sondagens possuem um alto custo envolvido, devido à movimentação e operação dos equipamentos de perfuração. O planejamento cuidadoso de quantidade, extensão e localização das sondagens é, portanto, fundamental para a minimização dos custos. Nesse sentido, o aprimoramento das abordagens para solução do problema pode ser de grande benefício, pois mesmo reduções de pequenos percentuais no custo trazem substancial economia, dada a magnitude dos valores envolvidos.

Por fim, uma metaheurística proposta no trabalho de Bilal *et al.* (2014), denominada Busca Tabu Iterada (*Iterated Tabu Search*, ITS), é investigada. É proposto o uso simultâneo de uma abordagem paralela e uma nova vizinhança a fim de melhorar a qualidade das soluções obtidas pelo método. O estudo se concentra no conjunto de instâncias encontrado na literatura (Bilal *et al.* 2014).

O trabalho está organizado da seguinte forma. Na Seção 2 o PRCA é apresentado. Uma aplicação para o problema é exposta na Seção 3, enquanto na Seção 4 uma metaheurística para solução do problema é abordada. Os resultados computacionais são reportados na Seção 5 e as conclusões na Seção 6.

2. Problema de recobrimento de conjuntos com agrupamento

O problema de recobrimento de conjuntos com agrupamento pode ser definido como uma variante do problema de recobrimento de conjuntos, na qual os conjuntos de recobrimento são particionados em K agrupamentos disjuntos. Para cada agrupamento há um custo fixo associado, que deve ser pago quando ao menos um conjunto de recobrimento nele contido é selecionado.

O PRCA é definido formalmente por Alfandari e Monnot (2014) da seguinte forma. Seja $C = \{1, \ldots, n\}$ um conjunto de elementos e $\mathcal{S} = \{S_1, \ldots, S_m\}$ uma coleção de subconjuntos de C. Cada subconjunto $S_j \in \mathcal{S}$ tem associado um custo positivo $c_j = c(S_j)$. Assume-se ainda que o conjunto de índices $J = \{1, \ldots, m\}$ é particionado em K subconjuntos disjuntos, de forma que $\mathcal{J} = \{J_k \colon k = 1, \ldots, K\}$, ou seja, $\bigcup_{k=1}^{K} J_k = J$ e $J_k \cap J_{k'} = \emptyset$ para todo $k \neq k'$. Cada agrupamento é definido por $\mathcal{F}_k = \{S_j \in \mathcal{S} \colon j \in J_k\}$ e um custo f_k é pago quando pelo menos um subconjunto contido em $\mathcal{F}_k$ é selecionado. O PRCA consiste em cobrir todos os elementos de C através da seleção de uma coleção de subconjuntos $\mathcal{S}' \subset \mathcal{S}$, minimizando $c(\mathcal{S}')$ e o custo dos agrupamentos utilizados, ou seja, $\overline{c}(\mathcal{S}') = c(\mathcal{S}') + \sum_{k \colon \mathcal{S}' \cap \mathcal{F}_k \neq \emptyset} f_k$. Seja o seguinte modelo matemático.

Parâmetros:

$$a_{ij} = \begin{cases} 1 & \text{se } i \in S_j \\ 0 & \text{caso contrário} \end{cases}$$

$$f_k = \text{custo do agrupamento } k.$$

$$c_j = \text{custo do conjunto } j.$$

Variáveis:

$$x_j = \begin{cases} 1 & \text{se o conjunto } S_j \text{ é selecionado} \\ 0 & \text{caso contrário} \end{cases}$$

$$y_k = \begin{cases} 1 & \text{se o agrupamento } k \text{ é selecionado} \\ 0 & \text{caso contrário} \end{cases}$$

Modelo matemático:

$$\text{Minimizar } \sum_{k=1}^{K} f_k y_k + \sum_{j=1}^{m} c_j x_j \tag{1}$$

Sujeito a:

$$\sum_{j=1}^{m} a_{ij} x_j \geq 1 \qquad i = 1, \ldots, n \tag{2}$$

$$y_k \geq x_j \qquad k = 1, \ldots, K,\ j \in J_k \tag{3}$$

$$x_j \in \{0, 1\} \qquad j = 1, \ldots, m \tag{4}$$

$$y_k \in \{0, 1\} \qquad k = 1, \ldots, K \tag{5}$$

A Função Objetivo (1) considera a minimização dos custos pagos pelos conjuntos e agrupamentos selecionados. As Restrições (2) estabelecem que cada elemento deve ser coberto por pelo menos um conjunto. Nas Restrições (3) é definido que se um agrupamento é selecionado, os conjuntos de cobertura contidos são liberados para seleção. Por fim, as Restrições (4) e (5) definem o domínio das variáveis.

De acordo com Alfandari e Monnot (2014), muitos problemas de transporte podem ser formulados como PRCA, como o problema de determinação de viagens abordado por Desaulniers *et al.* (1997) ou o problema de roteamento de veículos com frota heterogênea de Barnhart *et al.* (2003). Bilal *et al.* (2014) apresentam um problema que tem origem na indústria de mineração e que pode ser modelado como PRCA, conforme é mostrado a seguir.

3. Aplicação na indústria de mineração

Na indústria de mineração, a primeira etapa para o estabelecimento de minas de extração de minérios geralmente consiste em um estudo geológico com a finalidade de avaliar a viabilidade da região. Na etapa seguinte, são feitas longas perfurações de caráter exploratório, denominadas sondagens. Através das sondagens são obtidas amostras do solo, que servem para validar ou ajustar as estimativas iniciais da localização dos minérios. Como a perfuração do solo envolve alto custo, é importante otimizar a quantidade, localização, orientação e extensão das sondagens. Há ainda um custo associado à movimentação e posicionamento dos equipamentos que deve ser levado em consideração.

Os resultados do estudo costumam ser apresentados em um diagrama de blocos, no qual o solo é dividido em porções de formato cúbico e a cada porção é associada uma estimativa da quantidade de minerais contida. Ao representar no diagrama os locais onde os equipamentos de perfuração podem ser instalados, é possível determinar todas as sondagens factíveis. Um bloco é considerado coberto por uma sondagem se a distância ortogonal do seu centro até a perfuração é menor que um determinado valor. Um exemplo bidimensional do diagrama é apresentado na Figura 1.

A modelagem do problema subjacente como PRCA é bastante intuitiva. Nesse caso, os blocos correspondem aos elementos a serem cobertos, enquanto as sondagens são equivalentes aos conjuntos de recobrimento, uma vez que cada uma cobre um subconjunto de elementos. O custo da movimentação dos equipamentos é análogo ao custo fixo de um agrupamento, pois, ao ser pago, libera um grupo de conjuntos de recobrimento. A Figura 1 ilustra possíveis sondagens a partir de dois posicionamentos distintos para os equipamentos e auxilia na visualização da equivalência entre problemas.

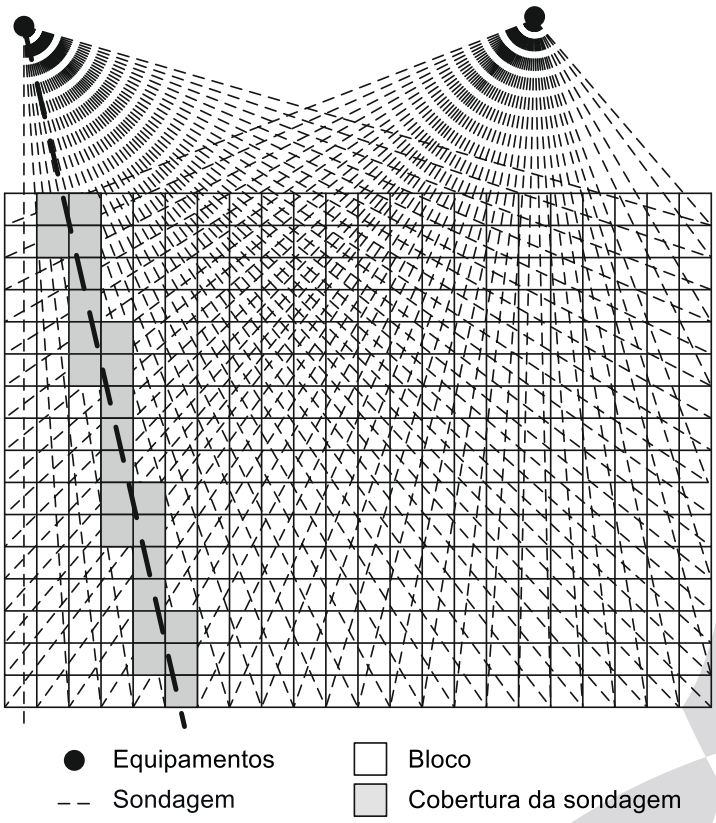

Figura 1: Representação de potenciais sondagens. Figura extraída de Bilal *et al.* (2014)

4. Método de resolução

Métodos de otimização combinatória normalmente são divididos em duas categorias: exatos e heurísticos. A principal característica de interesse em algoritmos exatos é a sua garantia de encontrar e provar a solução ótima para qualquer instância de problemas de otimização. Abordagens exatas, entretanto, costumam ser eficientes apenas para instâncias de pequeno e médio porte, visto que o tempo de execução costuma aumentar exponencialmente conforme a dimensão da instância, restringindo assim a aplicação prática destes métodos. São exemplos de abordagens exatas *branch-and-bound*, programação dinâmica e métodos baseados em programação linear e inteira, como *branch-and-cut*, *branch-and-price* e *branch-and-cut-and-price*.

Diferentemente dos algoritmos exatos, métodos heurísticos buscam apenas fornecer uma boa solução, não necessariamente ótima, em tempo reduzido. Como problemas de otimização reais podem ter grande dimensão, frequentemente, heurísticas ou metaheurísticas são as únicas opções viáveis. De maneira geral, espera-se que métodos heurísticos encontrem soluções próximas ao ótimo com alta frequência, tenham baixa probabilidade de obter soluções distantes do ótimo e demandem tempo moderado. Sob a designação de metaheurísticas pode-se citar como exemplo: *Simulated Annealing*, Busca Tabu, Busca em Vizinhança Variável, Algoritmos Genéticos e GRASP (Osman e Kelly 1996; Blum e Roli 2003).

Como o PRCA foi definido recentemente na literatura, foi encontrada apenas uma metaheurística projetada especificamente para o problema. O método é devido a Bilal *et al.* (2014) e consiste em uma combinação de Busca Local Iterada com Busca Tabu. Busca Local Iterada é uma metaheurística que realiza sucessivas iterações de Busca Local e que conta com um operador de perturbação que define um novo ponto inicial para a busca a cada iteração. Esta metaheurística tem sido aplicada a uma variedade de problemas como: Problema de Roteamento de Veículos e

variantes (Penna *et al.* 2013; Walker *et al.* 2012; Michallet *et al.* 2014), Problema de *Bin Packing* Multicapacitado (Masson *et al.* 2013) e Problema de Programação *Flow Shop* Permutacional (Dong *et al.* 2009; Marmion *et al.* 2011).

A Busca Tabu é um algoritmo que estende busca local, fazendo uso de estruturas de memória para evitar ciclos e escapar de ótimos locais. A memória consiste em uma lista de movimentos proibidos, chamada lista tabu, que é mantida de tal forma que, sempre que um movimento é executado, seu inverso é adicionado à lista. Um movimento fica proibido enquanto estiver na lista tabu e é liberado após um determinado número de iterações. Glover (1989) apresenta os princípios fundamentais subjacentes ao método.

No algoritmo resultante da combinação, que é denominado Busca Tabu Iterada, é feita a substituição do operador de Busca local da Busca Local Iterada por Busca Tabu. Para diversificação da solução são propostas duas funções de perturbação, sendo a que primeira diversifica a busca ao redor do ótimo local atual, enquanto a segunda tem por objetivo forçar grandes saltos no espaço de busca a fim de explorar novas vizinhanças.

Para avaliar o método apresentado, Bilal *et al.* (2014) adaptaram o algoritmo memético de Beasley e Chu (1996), originalmente proposto para resolução do PRC clássico. Os experimentos computacionais consideraram um conjunto de 30 instâncias, obtidas a partir de dados reais da indústria de mineração.

Duas contribuições deste trabalho se referem à metaheurística ITS. A primeira delas consiste em uma abordagem paralela para o componente de Busca Tabu, fundamentada no fato de a avaliação do melhor movimento não possuir dependência de dados e ser, portanto, facilmente paralelizável. A segunda contribuição compreende a hibridização da ITS, através da adição de uma vizinhança resolvida com programação inteira.

Nas seções seguintes são apresentadas as duas referidas contribuições para o trabalho de Bilal *et al.* (2014).

4.1. Abordagem paralela para a ITS

Com a atual alta disponibilidade de *hardware* com vários núcleos, a computação paralela tem recebido crescente atenção. Quando utilizadas corretamente, técnicas de paralelismo podem gerar significativa economia de tempo ou permitir a abordagem de grandes problemas até então considerados inviáveis. São exemplos de trabalhos recentes que relacionam otimização combinatória e computação paralela: Busca Tabu cooperativa multivizinhança paralela (Jin *et al.* 2012) e *Simulated Annealing* paralelo (Wang *et al.* 2015), ambos para problemas de roteamento de veículos, e um algoritmo genético de chaves aleatórias para um problema de telecomunicações (Andrade *et al.* 2015). Ainda, Crainic e Toulouse (2010) apresentam uma visão geral sobre estratégias e princípios de implementação paralelos aplicáveis a metaheurísticas.

Durante os testes de análise de desempenho conduzidos neste trabalho, foi observado que a maior parte do tempo de execução era dispendida na escolha do melhor movimento na Busca Tabu. Essa escolha consiste em um laço de repetição no qual é invocada a função de avaliação de movimentos para cada conjunto de recobrimento e é escolhido aquele que gerar o maior ganho na função objetivo. Se esse conjunto estiver selecionado na solução corrente, então o movimento executado efetuará sua remoção. Caso contrário, o conjunto será adicionado à solução.

A avaliação de adição ou remoção de um conjunto de recobrimento j depende apenas da leitura de informações imutáveis como o custo do próprio conjunto, o de seu agrupamento e os ganhos relativos aos elementos por ele cobertos. Portanto, como a avaliação de um movimento é completamente independente da avaliação dos demais, foi identificada uma oportunidade para paralelização da seleção do melhor movimento.

A abordagem implementada neste trabalho consistiu na paralelização da escolha do melhor movimento, de forma que múltiplas *threads* avaliem simultaneamente uma parcela dos conjuntos cada uma. Após todas as *threads* finalizarem, os resultados individuais são comparados e o melhor deles é escolhido.

Por não haver nenhuma dependência de dados e envolver um gargalo no algoritmo, a paralelização trouxe amplo ganho de desempenho. Com o uso de 12 *threads*, em algumas das instâncias de maiores dimensões foram observadas reduções superiores a 80% no tempo necessário para atingir o mesmo número de iterações da abordagem sem paralelismo. A ferramenta escolhida foi a API OpenMP.

4.2. Vizinhança HVF para a ITS

Dado o modelo matemático formalizado por Bilal *et al.* (2014), neste trabalho é definida uma nova vizinhança, que faz uso de programação inteira, e é proposta sua integração à metaheurística ITS. Nessa vizinhança é incorporada uma estratégia heurística baseada em *hard variable fixing* (HVF) ou *diving* (Bixby *et al.* 2000). A estratégia resume-se a resolver o modelo sucessivas vezes fixando parte das variáveis que correspondem aos conjuntos de recobrimento ativos, i.e., cujo valor na solução corrente é não nulo.

Conforme visto na seção anterior, o uso de paralelismo trouxe significativo ganho de desempenho. A ideia subjacente ao uso da vizinhança HVF é ocupar o tempo economizado com o polimento de soluções promissoras obtidas pela Busca Tabu. Uma solução é considerada promissora se o valor de sua função objetivo está dentro de um determinado *gap* em relação à melhor solução encontrada até então. A escolha de não empregar para todas as soluções se deve ao custo computacional relativamente alto da vizinhança HVF. O *gap* é calculado conforme segue.

$$gap = \frac{S_{incumbente} - S_{corrente}}{S_{incumbente}} \times 100$$

O Algoritmo 1 ilustra a vizinhança HVF, que é formalizada conforme segue.

Algoritmo 1 $vizinhancaHVF$

Entrada: S, n, r
Saída: Melhor solução encontrada S^*
$S^* \leftarrow S$
$fila \leftarrow$ uma fila contendo todos os conjuntos selecionados em S
$fila \leftarrow ordenaAleatorio(fila)$
$estagn \leftarrow 0$
Fixa todos os conjuntos da fila, exceto os n primeiros
Para $j = 1$ até r **e** $estagn < r/2$ **faça**
    $S \leftarrow resolveModelo()$
    Se $avalia(S) > avalia(S^*)$ **então**
        $S^* \leftarrow S$
        Libera todos os conjuntos fixados
        $fila \leftarrow$ fila atualizada contendo os conjuntos selecionados em S
        $fila \leftarrow ordenaAleatorio(fila)$
        Fixa todos os conjuntos da fila, exceto os n primeiros
        $estagn \leftarrow 0$
    Senão
        Fixa os n primeiros conjuntos da fila
        Move os n primeiros para o fim da fila
        Libera os novos n primeiros da fila
        $estagn \leftarrow estagn + 1$
    Fim Se
Fim Para
Libera todos os conjuntos fixados
Retorna S^*

Dada uma solução de entrada S, os conjuntos de recobrimento que estão selecionados são organizados em uma fila ordenada aleatoriamente. A seguir são fixados todos, exceto os primeiros n, conjuntos da fila. O modelo matemático é resolvido e a solução obtida é avaliada. Se houver melhora em relação à solução inicial S, todos os conjuntos fixados são liberados, a fila é atualizada

com os conjuntos de recobrimento da nova solução e eles são novamente fixados segundo o mesmo critério considerado anteriormente. Caso não haja melhora, os primeiros n elementos são fixados e movidos para o fim da fila, enquanto os novos primeiros n são liberados. Esse processo é repetido r vezes e a melhor solução encontrada é retornada. Dado o significativo custo computacional deste algoritmo, é considerado ainda um critério de parada adicional, que determina o encerramento do mesmo caso não haja melhora na solução por $r/2$ iterações consecutivas.

5. Resultados computacionais

Para facilitar a comparação de resultados, todos os testes computacionais descritos nesta seção utilizam o modelo matemático proposto por Bilal *et al.* (2014), ou seja, a função objetivo é de maximização.

Os testes foram realizados em um computador com dois processadores Intel Xeon E5-2697 v2 e 64 GB de memória principal. As implementações foram feitas em linguagem C++ e compiladas com o G++ v4.7. O paralelismo da Busca Tabu e do resolvedor CPLEX (versão 12.5) foi limitado em 12 *threads*.

5.1. Instâncias

As instâncias consideradas neste trabalho foram obtidas diretamente com Nehmé Bilal, um dos autores da metaheurística ITS. O autor relatou que em seu trabalho as instâncias passaram por um pré-processamento durante a leitura, de forma que conjuntos de recobrimento com custo maior que a soma dos ganhos de seus elementos são eliminados.

A Tabela 1 apresenta as dimensões das 30 instâncias após o pré-processamento. De acordo com o número de elementos, as instâncias são aqui divididas em três grupos: A e B são consideradas pequenas, C e D são médias, enquanto E e F são grandes.

Tabela 1: Dimensões das instâncias pré-processadas

Instância	Elementos	Conjuntos	Agrupamentos
A1	1000	3212	10
A2	1000	4060	10
A3	1000	6015	10
A4	1000	11900	10
A5	1000	76932	10
B1	1000	6362	20
B2	1000	9638	20
B3	1000	23654	20
B4	1000	36601	20
B5	1000	64087	20
C1	5192	5891	31
C2	5192	10651	31
C3	5192	21312	31
C4	5192	32516	31
C5	5192	56449	31
D1	5192	13303	62
D2	5192	24694	62
D3	5192	49034	62
D4	5192	75264	62
D5	5192	103760	62
E1	15000	8521	100
E2	15000	16264	100
E3	15000	30673	100
E4	15000	57508	100
E5	15000	67368	100
F1	15000	86308	100
F2	15000	105959	100
F3	15000	148104	100
F4	15000	239331	100
F5	15000	408690	100

5.2. Definição dos parâmetros

Para as instâncias médias, os resultados computacionais preliminares revelaram que as soluções encontradas continham menor quantidade de conjuntos selecionados em relação as demais. Dessa forma, um número menor de conjuntos é fixado durante a vizinhança HVF, ocasionando um aumento no tempo necessário para sua resolução. Para essas instâncias, notou-se ainda a ocorrência de soluções muito diferentes, mas com valor da função objetivo muito próximo. Por consequência, mais soluções dentro do *gap* para execução da vizinhança HVF são identificadas e mais tempo é dispendido com a vizinhança, fazendo com o número total de iterações da ITS fosse drasticamente reduzido. Posto isso, para manter uma configuração de parâmetros consistente para instâncias pequenas e médias, optou-se pelo acréscimo de 1800 segundos ao tempo utilizado originalmente por Bilal *et al.* (2014).

Ainda, durante os testes computacionais com as instâncias grandes, foi observado que uma quantia significativa de tempo era consumida na aplicação da vizinhança HVF as soluções iniciais, geralmente de baixa qualidade. Como o polimento dessas soluções não é imprescindível e, de maneira geral, atrasou a convergência do algoritmo, adotou-se para essas instâncias um número mínimo de iterações ($iniIters$) antes que a vizinhança passe a ser aplicada. O valor escolhido para $iniIters$ é igual a 500.

Os demais parâmetros foram definidos empiricamente conforme segue. Para as instâncias pequenas e médias, o valor de p (intensidade da perturbação 1), que não é especificado no trabalho original, foi obtido em contato com Nehmé Bilal e é igual a 0,3. Para os demais parâmetros é mantida a mesma configuração de Bilal *et al.* (2014).

Para as instâncias grandes, p com valor igual a 0,2 mostrou-se mais adequado. No tempo limite não houve alteração e são utilizados, portanto, 36000 segundos. Da mesma forma, os demais parâmetros da ITS mantém os valores originais.

No que concerne a vizinhança HVF, n e r têm valor igual a 5 e 10, respectivamente, para todas as instâncias. O *gap* máximo para a execução da vizinhança é igual a 1% para instâncias pequenas e médias, e 0,2% para as grandes.

5.3. Resultados da ITS com vizinhança HVF

A metodologia seguida para os testes computacionais é a mesma de Bilal *et al.* (2014). A ITS com vizinhança HVF foi executada cinco vezes para cada instância e a melhor, a pior e a solução média são apresentadas. Os resultados obtidos para as instâncias pequenas e médias são apresentados na Tabela 2.

As soluções destacadas em negrito são novas melhores soluções conhecidas obtidas neste trabalho. A coluna σ apresenta o desvio padrão, enquanto t_m informa o tempo médio necessário para obtenção da melhor solução. It_m e M_m apresentam as médias para, respectivamente, a quantia total de iterações do algoritmo e o número de vezes que a solução incumbente foi melhorada. Já HVF_m especifica o número médio de vezes que a vizinhança HVF foi capaz de melhorar a solução incumbente. P_m indica o número médio de vezes que foi detectada estagnação e a segunda função de perturbação foi acionada, enquanto PM_m apresenta o número médio de vezes em que o uso dessa função levou a uma melhora na solução. Por fim, T mostra o tempo limite utilizado para cada instância.

A Tabela 3 apresenta uma comparação aos resultados de Bilal *et al.* (2014). A medida utilizada é o desvio relativo percentual (DRP), que é calculado da seguinte forma.

$$DRP = \frac{S_{obtida} - S_{original}}{S_{original}} \times 100$$

A comparação mostra uma consistente melhora na qualidade das soluções obtidas. A solução média é melhorada em 16 das 20 instâncias, sendo que para três das quatro restantes o algoritmo já era capaz de chegar ao ótimo em todas as execuções (A2, C1 e C3). O mesmo comportamento é observado nas piores soluções. A melhor solução obtida foi melhorada em 11 instâncias,

Tabela 2: Resultados para as instâncias pequenas e médias

Inst.	Pior	Melhor	Média	σ	t_m (s)	It_m	M_m	HVF_m	P_m	PM_m	T (s)
A1	150368	150386	150378,8	9,9	393	26573,0	15,6	14,6	38,2	1,6	3600
A2	179973	179973	179973,0	0,0	1374	1731,6	4,6	4,6	3,0	0,8	3600
A3	155266	155266	155266,0	0,0	629	47157,4	19,6	19,0	67,6	1,2	3600
A4	156395	156540	156453,0	53,4	2378	13603,2	20,8	20,4	20,0	2,6	3600
A5	160235	**160767**	160438,8	239,9	3548	2972,2	20,4	20,2	4,2	1,0	5400
B1	152223	152335	152306,2	48,5	297	26114,2	9,0	8,6	22,2	0,4	3600
B2	155554	**155752**	155593,6	88,5	1445	10783,6	16,4	15,2	9,4	1,8	3600
B3	157337	**158176**	157556,0	354,0	2340	8113,8	19,0	19,0	7,2	1,6	3600
B4	158104	**158411**	158254,2	148,6	2922	3175,6	17,0	16,8	2,6	0,8	3600
B5	158379	**159581**	158966,4	450,9	1637	1129,6	12,6	12,4	1,0	0,0	3600
C1	246121	246121	246121,0	0,0	61	1336,2	5,4	5,4	1,0	0,0	3600
C2	247447	247455	247450,2	4,4	1504	506,0	7,4	7,2	1,0	0,4	3600
C3	249070	249070	249070,0	0,0	1091	221,2	7,0	7,0	0,2	0,0	3600
C4	249088	249124	249110,8	18,2	2375	150,8	7,6	7,4	0,0	0,0	3600
C5	249838	249935	249881,2	49,3	2812	119,4	9,6	9,6	0,0	0,0	5400
D1	247113	**247250**	247185,4	50,4	1275	1011,6	13,6	13,6	0,8	0,0	3600
D2	247935	248389	248087,4	180,3	1554	249	14,2	14,0	0,2	0,0	3600
D3	248483	**249165**	248818,4	338,6	3269	138,0	10,8	10,8	0,0	0,0	5400
D4	248353	248949	248691,0	222,9	7257	150,2	14,8	14,8	0,0	0,0	9000
D5	249860	**250288**	250019,2	165,0	7514	79,0	13,0	12,8	0,0	0,0	9000

Tabela 3: Desvio relativo percentual em relação aos resultados de Bilal *et al.* (2014) para instâncias pequenas e médias

Inst.	Pior(%)	Melhor(%)	Média(%)	$t_m(\%)$	$It_m(\%)$	$M_m(\%)$	$P_m(\%)$	$PM_m(\%)$	$T(\%)$
A1	**0,140**	0,000	**0,089**	147,17	30,10	6,85	27,33	-27,27	100
A2	0,000	0,000	0,000	1026,23	-81,48	-51,06	-78,57	-50,00	100
A3	**0,724**	0,000	**0,230**	1,58	384,66	-30,00	376,06	-33,33	100
A4	**0,097**	**0,075**	**0,073**	395,21	180,65	-11,11	194,12	160,00	100
A5	**0,352**	**0,306**	**0,297**	210,86	69,74	-35,44	90,91	66,67	50
B1	**0,074**	0,000	**0,011**	272,61	165,66	-42,31	141,30	-75,00	100
B2	**0,093**	**0,127**	**0,045**	419,41	62,97	-8,89	56,67	80,00	100
B3	**0,056**	**0,397**	**0,064**	158,63	235,68	-20,17	157,14	-11,11	100
B4	**0,147**	**0,005**	**0,025**	197,07	85,60	-21,30	8,33	-42,86	100
B5	**0,051**	**0,097**	**0,088**	101,95	9,27	-37,62	0,00	-100,00	100
C1	0,000	0,000	0,000	490,38	-75,33	-71,87	-66,67	0,00	100
C2	0,000	0,000	0,000	323,31	-81,76	-54,32	-70,59	-81,00	100
C3	0,000	0,000	0,000	1066,03	-86,76	-71,77	-90,00	-100,00	100
C4	**0,011**	**0,012**	**0,015**	336,78	-89,54	-65,77	-100,00	-100,00	100
C5	**0,135**	**0,022**	**0,050**	59,48	-93,94	-53,40	-100,00	-100,00	50
D1	**0,002**	**0,021**	**0,006**	117,84	-63,41	-50,36	-76,47	-100,00	100
D2	**0,086**	0,000	**0,048**	57,78	-81,09	-54,78	-80,00	-100,00	100
D3	**0,004**	**0,097**	**0,062**	32,44	-88,72	-57,81	-100,00	-100,00	50
D4	**0,099**	-0,063	**0,008**	34,99	-89,30	-57,47	-100,00	-100,00	25
D5	**0,102**	**0,052**	**0,025**	118,46	-94,43	-58,86	-100,00	-100,00	25

sendo que para 8 obteve-se uma nova melhor solução conhecida. Em D4 nota-se uma exceção, onde a melhor solução é 0,063% pior do que a de Bilal *et al.* (2014). Em negrito são destacados os casos onde houve melhora na qualidade da solução.

Na coluna t_m observa-se que, para as instâncias pequenas e médias, a adição da vizinhança HVF leva a um aumento significativo, em geral, no tempo para obter a melhor solução. O número de iterações é superior para as instâncias A e B (A2 é exceção) e inferior para C e D. Conforme mencionado anteriormente, características intrínsecas a C e D fazem com que a resolução da vizinhança HVF tome mais tempo, reduzindo o total de iterações. Contudo, apesar dessas significativas reduções, foi possível melhorar a qualidade das soluções. Ainda, para as instâncias onde houve redução do total de iterações, observa-se, conforme esperado, uma redução no número de chamadas à segunda função de perturbação. Para as demais nota-se que a perturbação foi ativada mais vezes. Em 5 instâncias, a função nunca foi utilizada. Na coluna PM_m é possível observar que, quando utilizada, houve grande variação quanto ao número de vezes que a perturbação foi capaz de melhorar a solução. Já em M_m é visto que, em geral, o polimento da solução reduziu o número de vezes que a incumbente é atualizada. Por fim, T apresenta valores maiores que zero, pois o tempo limite utilizado é maior que o original.

Na Tabela 4 são apresentados os resultados para as instâncias grandes.

Tabela 4: Resultados para as instâncias grandes

Inst.	Pior	Melhor	Média	σ	t_m (s)	It_m	M_m	MA_m	MV_m	P_m	PM_m
E1	620895	**623606**	621962,0	1018,4	17717	59752,0	55,4	23,8	14,4	195,8	2,8
E2	448420	449257	448898,8	363,2	23195	70043,8	61,6	28,0	12,6	229,8	2,4
E3	547653	**551531**	549323,0	1554,7	28508	40480,8	63,0	37,2	22,6	129,2	3,0
E4	635868	**637825**	636640,4	742,0	24750	4967,6	63,4	26,4	25,6	11,6	1,2
E5	470445	**474222**	471652,8	1524,9	27757	43478,8	62,6	25,8	19,2	139,0	1,6
F1	499462	**501319**	500452,8	827,2	27000	13180,6	79,8	39,4	35,8	37,6	1,8
F2	493710	**495887**	494511,6	986,0	29901	23419,8	82,4	43,6	42,4	70,8	3,0
F3	483929	**485865**	484909,0	715,5	25524	12011,8	57,8	22,8	21,2	34,6	1,6
F4	482359	**485126**	484126,4	1121,9	18435	8305,8	44,0	10,8	10,0	24,6	0,6
F5	495644	**498559**	496947,2	1433,6	16954	3563,0	36,0	5,2	5,2	9,4	0,0

Para 9 das 10 instâncias foram conhecidas novas melhores soluções, que estão destacadas em negrito.

Conforme mencionado anteriormente, para as instâncias grandes foi adotado um número mínimo de iterações necessárias para que a vizinhança HVF passe a ser aplicada. Dessa forma, na Tabela 4 as colunas MA_m e MV_m indicam, respectivamente, o número médio de vezes que a solução incumbente foi atualizada após as iterações iniciais, e o número médio de vezes em que a vizinhança foi capaz de melhorar a solução incumbente. As demais colunas apresentam as mesmas informações descritas para as tabelas anteriores.

A comparação aos resultados de Bilal *et al.* (2014) é apresentada na Tabela 5.

Em negrito são destacados os resultados para os quais houve melhora na qualidade da solução. A pior solução e solução média são melhores para todas as instâncias, enquanto a melhor solução é pior apenas para E2 (e apenas 0,007% pior).

Na coluna t_m é observado um aumento para as 3 primeiras instâncias, enquanto há redução para as demais. Quanto ao total de iterações, apenas E4 teve redução. Da mesma forma, o número de chamadas à segunda função de perturbação é menor apenas para E4. Embora a função, em geral, tenha sido utilizada mais vezes, foi observada redução em PM_m para 7 instâncias. Na coluna M_m é visto que, à exceção de E2, o número de atualizações da função incumbente foi superior. É possível que isso seja consequência da opção pela redução do valor do parâmetro p, que gera mudanças mais sutis nas soluções a cada iteração e faz com que a convergência se dê através de passos menores. Por fim, o mesmo tempo de Bilal *et al.* (2014) foi utilizado e o DRP, portanto, é igual a zero para as

Tabela 5: Desvio relativo percentual em relação aos resultados de Bilal *et al.* (2014) para instâncias grandes

Inst.	Pior(%)	Melhor(%)	Média(%)	$t_m(\%)$	$It_m(\%)$	$M_m(\%)$	$P_m(\%)$	$PM_m(\%)$
E1	**0,591**	**0,614**	**0,503**	38,34	5,18	60,12	5,04	133,33
E2	**0,639**	-0,007	**0,430**	45,28	13,62	-2,53	16,77	-14,29
E3	**0,701**	**0,624**	**0,745**	93,27	73,12	42,53	81,46	150,00
E4	**0,090**	**0,128**	**0,090**	-13,42	-46,70	37,83	-57,66	-85,71
E5	**0,436**	**0,437**	**0,148**	-3,60	258,82	32,07	323,78	0,00
F1	**0,656**	**0,104**	**0,294**	-13,03	51,19	78,12	50,40	-73,53
F2	**1,245**	**0,314**	**0,797**	-8,49	184,22	92,52	195,00	-59,46
F3	**0,465**	**0,115**	**0,358**	-26,27	125,87	45,96	147,14	-76,47
F4	**0,486**	**0,060**	**0,489**	-29,94	273,80	69,23	355,56	-82,35
F5	**0,636**	**0,706**	**0,595**	-51,14	150,21	80,00	193,75	-100,00

instâncias grandes.

6. Conclusões

Este trabalho abordou o PRCA, uma variante do PRC clássico na qual os conjuntos de recobrimento são organizados em K agrupamentos disjuntos. Sobre cada agrupamento incide um custo fixo a ser pago uma única vez sempre que ao menos um conjunto contido é selecionado.

Foi apresentado um problema encontrado na literatura que é uma aplicação prática para o PRCA. O problema é devido a Bilal *et al.* (2014) e é oriundo da indústria de mineração, os autores propoẽm um conjunto de instâncias e uma metaheurística baseada em ITS, que guiaram os experimentos computacionais desenvolvidos neste trabalho.

Duas contribuições apresentadas neste trabalho se referem ao método ITS. A primeira consistiu na paralelização do componente de Busca Tabu, enquanto a segunda residiu na adição de uma nova vizinhança, resolvida com programação inteira, e que baseia-se em *hard variable fixing*.

Nos testes computacionais, a metaheurística foi executada 5 vezes para cada instância e a pior solução, melhor solução e solução média foram apresentadas. Os resultados mostraram que as contribuições ao método ITS proporcionaram melhora na qualidade das soluções para a ampla maioria das instâncias. Observando a solução média e a pior solução, notou-se melhora em 26 das 30 instâncias, sendo que para 3 das demais o algoritmo já era capaz de sempre chegar ao ótimo. Quanto à melhor solução, foi possível observar melhoria para 20 instâncias, sendo que os resultados de 17 são novas melhores soluções conhecidas. É importante ressaltar que, embora os percentuais de melhora pareçam pequenos, quando os valores são tomados em absoluto, as melhoras são bastante significativas, principalmente para as instâncias grandes. Visto que os custos envolvidos no problema são de grande magnitude, a utilização das estratégias propostas neste trabalho podem ser de grande benefício econômico.

Referências

Alfandari, Laurent e Jérôme Monnot (2014). "A note on the Clustered Set Covering Problem". Em: *Discrete Applied Mathematics* 164, pp. 13–19. ISSN: 0166218X.

Andrade, Carlos E. *et al.* (2015). "A biased random-key genetic algorithm for wireless backhaul network design". Em: *Applied Soft Computing* 33.0, pp. 150 –169. ISSN: 1568-4946.

Barnhart, Cynthia, Peter Belobaba e Amedeo R. Odoni (2003). "Applications of Operations Research in the Air Transport Industry". Em: *Transportation Science* 37.4, pp. 368–391.

Beasley, J.E e P.C Chu (1996). "A genetic algorithm for the set covering problem". Em: *European Journal of Operational Research* 94.2, pp. 392 –404. ISSN: 0377-2217.

Bilal, Nehme, Philippe Galinier e Francois Guibault (2014). "An iterated-tabu-search heuristic for a variant of the partial set covering problem". Em: *Journal of Heuristics* 20.2, pp. 143–164. ISSN: 1381-1231.

Bixby, E.Robert *et al.* (2000). "MIP: Theory and Practice — Closing the Gap". English. Em: *System Modelling and Optimization*. Ed. por M.J.D. Powell e S. Scholtes. Vol. 46. IFIP — The International Federation for Information Processing. Springer US, pp. 19–49. ISBN: 978-1-4757-6673-8.

Blum, Christian e Andrea Roli (2003). "Metaheuristics in combinatorial optimization: Overview and conceptual comparison". Em: *ACM Computing Surveys (CSUR)* 35.3, pp. 268–308.

Crainic, Teodor Gabriel e Michel Toulouse (2010). "Parallel Meta-heuristics". English. Em: *Handbook of Metaheuristics*. Ed. por Michel Gendreau e Jean-Yves Potvin. Vol. 146. International Series in Operations Research & Management Science. Springer US, pp. 497–541. ISBN: 978-1-4419-1663-1.

Desaulniers, G. *et al.* (1997). "Crew pairing at Air France". Em: *European Journal of Operational Research* 97.2, pp. 245 –259. ISSN: 0377-2217.

Dong, Xingye, Houkuan Huang e Ping Chen (2009). "An iterated local search algorithm for the permutation flowshop problem with total flowtime criterion". Em: *Computers & Operations Research* 36.5. Selected papers presented at the Tenth International Symposium on Locational Decisions (ISOLDE X), pp. 1664 –1669. ISSN: 0305-0548.

Glover, Fred (1989). "Tabu Search—Part I". Em: *ORSA Journal on Computing* 1.3, pp. 190–206.

Jin, Jianyong, Teodor Gabriel Crainic e Arne Løkketangen (2012). "A parallel multi-neighborhood cooperative tabu search for capacitated vehicle routing problems". Em: *European Journal of Operational Research* 222.3, pp. 441 –451. ISSN: 0377-2217.

Marmion, Marie-Eléonore *et al.* (2011). "NILS: A Neutrality-Based Iterated Local Search and Its Application to Flowshop Scheduling". English. Em: *Evolutionary Computation in Combinatorial Optimization*. Ed. por Peter Merz e Jin-Kao Hao. Vol. 6622. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 191–202. ISBN: 978-3-642-20363-3.

Masson, Renaud *et al.* (2013). "An iterated local search heuristic for multi-capacity bin packing and machine reassignment problems". Em: *Expert Systems with Applications* 40.13, pp. 5266 –5275. ISSN: 0957-4174.

Michallet, Julien *et al.* (2014). "Multi-start iterated local search for the periodic vehicle routing problem with time windows and time spread constraints on services". Em: *Computers & Operations Research* 41.0, pp. 196 –207. ISSN: 0305-0548.

Osman, Ibrahim H. e James P. Kelly (1996). *Meta-Heuristics: Theory and Applications*. Norwell, MA, USA: Kluwer Academic Publishers. ISBN: 0792397002.

Penna, Puca H. V., Anand Subramanian e Luiz S. Ochi (2013). "An Iterated Local Search heuristic for the Heterogeneous Fleet Vehicle Routing Problem". English. Em: *Journal of Heuristics* 19.2, pp. 201–232. ISSN: 1381-1231.

Walker, James D. *et al.* (2012). "Vehicle Routing and Adaptive Iterated Local Search within the HyFlex Hyper-heuristic Framework". English. Em: *Learning and Intelligent Optimization*. Ed. por Youssef Hamadi e Marc Schoenauer. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 265–276. ISBN: 978-3-642-34412-1.

Wang, Chao *et al.* (2015). "A parallel simulated annealing method for the vehicle routing problem with simultaneous pickup–delivery and time windows". Em: *Computers & Industrial Engineering* 83.0, pp. 111 –122. ISSN: 0360-8352.