Resolução do problema do conjunto de vértices de retroalimentação de peso mínimo como uma floresta induzida de peso máximo

Michell F. Queiroz e Rafael A. Melo
Computational Intelligence and Optimization Research Lab (CInO)
Departamento de Ciência da Computação, Universidade Federal da Bahia
Av. Adhemar de Barros, s/n, Salvador, BA 40170-110, Brasil
{michellfelipe,melo}@dcc.ufba.br

RESUMO

Dado um grafo simples não-direcionado $G = (V, E)$ com pesos associados aos vértices, o problema do conjunto de vértices de retroalimentação de peso mínimo (MWFVS, do inglês *minimum weighted feedback vertex set*) consiste em encontrar um subconjunto dos vértices $F \subseteq V$ de peso mínimo cuja remoção torne o grafo acíclico. No presente trabalho, aborda-se este problema NP-difícil por meio da resolução do problema da floresta induzida de peso máximo (MWIF, do inglês *maximum weighted induced forest*). São propostas: (a) uma nova formulação compacta de programação inteira mista, i.e., com quantidades polinomiais de variáveis e restrições; e (b) uma metaheurística *multi-start* baseada em busca local iterativa (ILS, do inglês *interated local search*). Resultados computacionais preliminares mostram que as técnicas propostas são competitivas com as melhores disponíveis na literatura, sendo capaz de superar as últimas em certos casos. O trabalho foi desenvolvido pelo aluno sob supervisão do orientador.

PALAVRAS CHAVE. Conjunto de vértices de retroalimentação, Programação inteira mista, Metaheurísticas.

Tópicos (MH - Metaheuristicas, OC - Otimização Combinatória, TAG - Teoria e Algoritmos em Grafos)

ABSTRACT

Given a vertex weighted undirected simple graph $G = (V, E)$, the minimum weighted feedback vertex set problem (MWFVS) consists in obtaining a subset of the vertices $F \subseteq V$ with minimum weight whose removal turns the graph acyclic. In this work, we approach this NP-hard problem by tackling the maximum weighted induced forest problem (MWIF). We propose: (a) a new compact mixed integer programming formulation, i.e., with polynomial quantities of variables and constraints; and (b) a multi-start iterated local search metaheuristic. Preliminary computational results show that the proposed techniques are competitive with the best ones available in the literature, being able to outperform the latter in certain cases. This work was developed by the student under supervision of his advisor.

KEYWORDS. Feedback vertex set, Mixed integer programming, Metaheuristics.

Paper topics (MH - Metaheuristics, OC - Combinatorial Optimization, TAG - Graph Theory and Algorithms)

1. Introdução

Dado um grafo simples não-direcionado $G = (V, E)$ com pesos associados aos vértices, um subconjunto dos vértices $F \subseteq V$ é denominado um conjunto de vértices de retroalimentação (FVS, do inglês *feedback vertex set*) se qualquer ciclo em G é formado por pelo menos um vértice $v \in F$. O problema do conjunto de vértices de retroalimentação de peso mínimo (MWFVS, do inglês *minimum weighted feedback vertex set*) consiste em encontrar um conjunto de vértices de retroalimentação $F \subseteq V$ de peso mínimo. Karp [1972] demonstrou que a versão de decisão do problema é NP-completo através de uma redução a partir do problema da cobertura de vértices implicando que o MWFVS é NP-difícil.

O MWFVS possui uma diversidade de aplicações práticas em várias áreas. Wang et al. [1985] consideram o problema no contexto de prevenção e/ou remoção de *deadlocks* em sistemas operacionais . Outras aplicações incluem o estudo de monopólios em sistemas distribuídos síncronos (veja Peleg [1997] e Peleg [1998]), e problemas de satisfação de restrições e inferência bayesiana (Bar-Yehuda et al. [1998a]). Uma revisão da literatura relacionada a problemas de conjuntos de vértices e arestas de retroalimentação pode ser encontrada em Festa et al. [1999].

Embora o MWFVS seja NP-difícil, há topologias específicas de grafos para as quais algoritmos polinomiais são conhecidos. Um exemplo que será usado no presente trabalho foi estudado em Carrabs et al. [2004], onde os autores apresentam um algoritmo baseado em programação dinâmica com tempo de execução linear para grafos denominados diamantes pelos mesmos. A classe dos grafos diamantes dos autores é representada por grafos $G = (V, E)$ com ápices r e z, onde cada vértice $v \in V$ está presente em pelo menos um caminho entre r e z, e a remoção do vértice z de V faz de G uma árvore.

Alguns poucos trabalhos tratam de métodos exatos baseados em programação inteira para o MWFVS. Brunetta et al. [2000] abordam o problema utilisando um algoritmo *branch-and-cut* e uma heurística de busca local baseada em *Tabu Search*. A abordagem dos autores consiste em um conjunto de procedimentos heurísticos para a separação de desigualdades violadas pertencentes a uma descrição parcial do politopo do MWFVS e uma busca tabu para obtenção de soluções viáveis e de bons limites superiores para a função objetivo. As técnicas propostas pelos autores foram capazes de resolver até a otimalidade instâncias aleatórias e geométricas com até 60 vértices e instâncias planares com até 300 vértices. Funke e Reinelt [1996] realizaram um estudo poliedral do MWFVS no contexto de grafos direcionados.

Nos últimos anos, diferentes heurísticas foram propostas para o MWFVS. Carrabs et al. [2011] propõem uma busca tabu iterativa que explora eficientemente a vizinhança definida resolvendo-se o MWFVS em tempo polinomial nos grafos diamante. Carrabs et al. [2014] propuseram um procedimento guloso randomizado e um algoritmo memético combinando um algoritmo genético e busca local baseada na resolução do MWFVS em grafos diamante. O algoritmo proposto mostrou-se mais efetivo que outras heurísticas propostas na literatura. Diversos autores também desenvolveram algoritmos aproximativos para o problema (Bafna et al. [1999]; Bar-Yehuda et al. [1998b]; Kleinberg e Kumar [2001]).

Até onde se tem conhecimento, não há uma formulação compacta para o MWFVS na literatura. Neste trabalho propõem-se (a) uma nova formulação de programação inteira mista compacta para o problema bem como (b) uma heurística *multi-start* baseada em busca local iterativa. O restante do trabalho é organizado da seguinte forma. A Seção 2 formaliza a definição do MWFVS e apresenta uma formulação de programação inteira mista. Na Seção 3 são apresentados um algoritmo construtivo randomizado e o procedimento de busca local iterativa. Na Seção 4, são reportados experimentos computacionais preliminares baseados em um subconjunto das instâncias disponíveis na literatura. Comentários finais e propostas de trabalho futuro são discutidos na Seção 5.

2. Definição do problema e formulação de programação inteira mista

Seja $G = (V, E)$ um grafo simples não-direcionado com conjunto de vértices V e conjunto de arestas E, no qual existe um peso não-negativo w_v associado a cada vértice $v \in V$. Um subconjunto dos vértices $F \subseteq V$, com complemento $\bar{F} = V \setminus F$, é definido como um conjunto de vértices de retroalimentação (FVS) caso o subgrafo $G[\bar{F}]$ induzido por $\bar{F}$ seja acíclico, onde $G[\bar{F}] \subseteq G$ é o subgrafo com conjunto de vértices $\bar{F}$ contendo toda aresta $e \in E$ incidente a dois vértices em $\bar{F}$. Note que $G[\bar{F}]$ é uma floresta. Define-se o peso de um conjunto de vértices $F \subseteq V$ como $W[F] = \sum_{v \in F} w_v$. O problema do conjunto de vértices de retroalimentação de peso mínimo (MWFVS) consiste em encontrar um FVS F de peso mínimo, i.e. $W[F] \leq W[F']$ para todo FVS $F' \subseteq V$. Observe que se F é um FVS então $G[\bar{F}]$ é uma floresta induzida por $\bar{F}$, e $W[V] = W[F] + W[\bar{F}]$. Isso implica que resolver o MWFVS é equivalente a resolver o problema da floresta induzida de peso máximo (MWIF, do inglês *maximum weighted induced forest*). Desta forma, no restante deste trabalho, considera-se o problema de encontrar uma floresta induzida de peso máximo.

Com o intuito de obter uma formulação compacta de programação inteira mista baseada em fluxos para o MWIF, será definida uma pequena modificação no grafo de entrada tal que o problema consistirá em encontrar uma árvore no grafo modificado. Dado o grafo de entrada $G = (V, E)$, define-se o grafo $G_s = (V_s, E_s)$ no qual $V_s = V \cup \{s\}$, $E_s = E \cup \{(s, v) \mid v \in V\}$, e $w_s = 0$. O propósito do vértice s na formulação será interligar as diferentes componentes conexas da floresta induzida $G[\bar{F}]$.

A formulação de programação inteira mista considera uma versão direcionada do grafo G_s, i.e. para cada aresta $(u, v) \in E_s$ associam-se arcos (u, v) e (v, u). O novo conjunto formado por esses arcos é expressado por E'. Os conjuntos de arcos saindo e entrando de v são denotados por $A^+(v)$ e $A^-(v)$, respectivamente. Enviando-se uma unidade de fluxo a partir do vértice fonte s para cada vértice $v \in \bar{F}$ pode-se encontrar uma árvore conectando os vértices em $\bar{F} \cup \{s\}$. Definem-se as variáveis de decisão:

$$x_{uv} = \begin{cases} 1, & \text{se o arco } (u,v) \in E' \text{ é selecionado,} \\ 0, & \text{caso contrário;} \end{cases}$$

$$y_v = \begin{cases} 1, & \text{se o vértice } v \in V \text{ pertence à árvore,} \\ 0, & \text{caso contrário.} \end{cases}$$

Usando as variáveis descritas, o problema pode ser formulado como

$$z_{MWIF} = \max \sum_{v \in V} w_v y_v \tag{1}$$

$$\sum_{(u,v) \in A^-(v)} x_{uv} = y_v \quad \forall\, v \in V, \tag{2}$$

$$\sum_{(s,v) \in A^+(s)} f_{sv} = \sum_{v \in V} y_v, \tag{3}$$

$$\sum_{(u,v) \in A^-(v)} f_{uv} - \sum_{(v,u) \in A^+(v)} f_{vu} = -y_v, \quad \forall\, v \in V, \tag{4}$$

$$x_{uv} \leq f_{uv} \leq (|V| - 1) x_{uv} \quad \forall\, (u,v) \in E', \tag{5}$$

$$x_{uv} + x_{vu} \leq y_v \quad \forall\, v \in V,\ (u,v) \in E', \tag{6}$$

$$x_{uv} + x_{vu} \geq y_u + y_v - 1 \quad \forall\, u, v \in V, (u,v) \in E', \tag{7}$$

$$f_{uv} \geq 0 \quad \forall\, (u,v) \in E', \tag{8}$$

$$x \in \{0,1\}^{|E'|}, \tag{9}$$

$$y \in \{0,1\}^{|V|}. \tag{10}$$

A função objetivo (1) maximiza a soma total dos pesos dos vértices presentes na floresta. As restrições (2) determinam que todo vértice $v \in \bar{F}$ possui exatamente um arco de entrada. As restrições (3) e (4) têm por objetivo assegurar a conectividade e balancear o fluxo de cada vértice. As restrições (5) apenas vinculam as variáveis de fluxo f e as variáveis x, onde a quantidade de fluxo que flui por um arco é sempre menor ou igual ao número máximo de arestas em uma árvore, i.e, número total de vértices menos um. As restrições (6) e (7) definem que a solução é um grafo induzido $G[\bar{F}]$. As restrições (8), (9) e (10) asseguram a não-negatividade e integralidade das variáveis.

3. Metaheurística *multi-start* baseada em busca local iterativa

3.1. Heurística gulosa randomizada

Nesta seção, serão apresentados dois critérios gulosos para a seleção de um novo elemento a ser adicionado a uma solução obtida de maneira construtiva. Adicionalmente, será proposta uma heurística construtiva gulosa randomizada para a obtenção de soluções.

Considere as seguintes funções gulosas para seleção de um vértice $v \in V$ ainda não presente na solução que buscam maximizar o peso da floresta: (a) maximização do peso do vértice $p_1 = \max w_v$; e (b) maximização do peso ponderado pela quantidade de arestas incidentes $p_2 = \max w_v/d_G(v)$. Defina $Q \subseteq V$ como um conjunto de vértices não presentes na solução para os quais não foi realizada uma tentativa de adicioná-los à floresta induzida. Dados os vértices em Q e uma função gulosa, p_1 ou p_2, a lista retrita de candidatos (LRC) é definida como um subconjunto dos vértices contendo os melhores candidatos.

A heurística gulosa randomizada é descrita em Algoritmo 1. O conjunto de vértices que forma a floresta é denotado por S. O conjunto Q é inicializado com V e S é igual a um conjunto vazio (linhas 1-2). O conjunto Q é ordenado de acordo com a função gulosa escolhida (linha 3). A cada iteração do laço 4-9 o algoritmo seleciona aleatoriamente um vértice v presente na LRC e adiciona-o a S caso sua inserção em S não gere ciclos em $G[S]$, e posteriormente v é removido de Q. Esse processo é repetido até que Q esteja vazio.

Algoritmo 1: CONSTRUTIVO-RANDOMIZADO $(G = (V, E), w)$

1 $Q \leftarrow V$;
2 $S \leftarrow \varnothing$;
3 Ordene Q de acordo com um dos critérios de escolha gulosa citados;
4 **while** $Q \neq \varnothing$ **do**
5   Considere uma LRC com os melhores elementos de Q;
6   Selecione aleatoriamente um vértice v na LRC;
7   **if** $G[S \cup \{v\}]$ *não contém ciclos* **then**
8         $S \leftarrow S \cup \{v\}$;
9     $Q \leftarrow Q \setminus \{v\}$;
10 **return** S;

3.2. Busca local iterativa

O algoritmo proposto nesta seção, chamado de MULTISTART-ILS, consiste de uma heurística *multi-start* baseada em uma busca local iterativa e é descrito em Algoritmo 2. O algoritmo executa It_{max} iterações (linhas 1-8) e em cada uma delas uma solução inicial S_i é gerada com o algoritmo construtivo randomizado apresentado na Seção 3.1 (linha 2). Partindo dessa solução, efetua-se uma busca local baseada em grafos diamante (linha 3). Em BUSCA-LOCAL-DIAMANTE explora-se vizinhanças adicionando à solução um vértice $v \in \bar{S}_i$ e solucionando o problema do MWFVS em um grafo diamante. O algoritmo com tempo de execução linear baseado em programação dinâmica utilizado em BUSCA-LOCAL-DIAMANTE é apresentado em Carrabs et al. [2004]. Duas variações para a busca local foram consideradas: (a) primeiro aprimorante e (b) melhor aprimorante. Assim que uma solução fica presa em um ótimo local, ou seja, nenhum dos métodos da busca local é capaz de melhorá-la, um mecanismo de perturbação é acionado (linhas 4-8). Em PERTURBAÇÃO, k vértices são aleatoriamente removidos da solução S_i^* e então o método CONSTRUTIVO-RANDOMIZADO é novamente aplicado desconsiderando os vértices removidos. Após todos os vértices presentes no FVS serem testados, os vértices removidos são reinseridos caso não gerem nenhum ciclo na floresta. A solução é então novamente melhorada aplicando BUSCA-LOCAL-DIAMANTE. Por fim, seleciona-se a floresta induzida de maior peso (linha 9). Como última etapa, efetua-se uma busca local em grande vizinhança MIP-LNS (linha 10), utilizando-se a formulação (1)-(10) na qual $\alpha\%$ dos vértices na floresta são fixados e resolve-se um subproblema restrito de programação inteira mista.

Algoritmo 2: MULTISTART-ILS $(G = (V, E), w)$

1 **for** $i \leftarrow 1, ..., It_{max}$ **do**
2     $S_i \leftarrow$ CONSTRUTIVO-RANDOMIZADO (G) ;
3     $S_i^* \leftarrow$ BUSCA-LOCAL-DIAMANTE (G, S_i);
4   **for** $j \leftarrow 1, ..., ILS_{max}$ **do**
5         $S_i' \leftarrow$ PERTURBAÇÃO(k, G, S_i^*) ;
6         $S_i'^* \leftarrow$ BUSCA-LOCAL-DIAMANTE (G, S_i');
7   **if** $W[S_i'^*] > W[S_i^*]$ **then**
8             $S_i^* \leftarrow S_i'^*$;
9 $S^* \leftarrow \text{argmax}_{i=1,...It_{max}} W[S_i^*]$;
10 $S^* \leftarrow$ MIP-LNS (S^*, α);
11 **return** $G[S^*]$;

4. Experimentos computacionais preliminares

Nesta seção são apresentados alguns experimentos computacionais analisando o desempenho das técnicas propostas. Todos os experimentos foram realizados em uma máquina executando Xubuntu x86 64 GNU/Linux, com processador Intel Core i7-4770S 3.10GHz, 8Gb de RAM. Todos os algoritmos foram implementados em C++ e a formulação resolvida utilizando CPLEX 12.7.

Os testes preliminares foram realizados em um subconjunto das instâncias propostas em Carrabs et al. [2011] com o objetivo de verificar o comportamento dos métodos propostos e definir parâmetros antes das execuções completas para todas as instâncias. Esse conjunto inclui grafos dos tipos randômico, grade, toroidal e hipercubo. As instâncias são caracterizadas pelo número de vértices, número de arestas e um intervalo de pesos para os vértices. As instâncias são divididas em pequenas e grandes, sendo que as pequenas possuem 25, 50 ou 75 vértices, enquanto as grandes possuem entre 100 e 500 vértices. O intervalo de pesos associado aos vértices é um inteiro positivo que pode variar em três intervalos distintos: [10,25], [10,50] e [10,75]. Tentou-se selecionar um subconjunto com características diversas que fosse representativa do conjunto completo de instâncias. As instâncias são identificadas como $D\_Xy\_n\_m\_low\_up$. D expressa o tamanho das instâncias (S para pequenas e L para grandes). X representa o tipo de grafo: randômico (R), grade quadrado (SG), grade não-quadrado (NG), hipercubo (H) e toroidal (T). Os outros parâmetros caracterizam um identificador para a instância (y), número de vértices (n), número de arestas (m), e os limites mínimo (low) e máximo (up) dos pesos. Para os grafos grade e toroidal, n e m indicam o número de linhas e colunas da grade, respectivamente.

Nas execuções preliminares reportadas nesta seção, utilizaram-se os seguintes parâmetros. A LCR da heurística construtiva randomizada é composta dos dez melhores candidatos. Os números máximos de iterações e perturbações da heurística *multi-start* foram, respectivamente, $It_{max} = 10$ e $ILS_{max} = 400$. A busca local diamante foi executada utilizando-se uma abordagem melhor aprimorante. A busca local em grande vizinhança foi realizada fixando-se os 95% dos vértices de maior peso presentes na melhor solução encontrada nas etapas anteriores, com um tempo limite de 60 segundos. No fim do processo, a formulação foi executada com a melhor solução encontrada pela metaheurística como *warm start*, por um tempo máximo de 600 segundos.

4.1. Instâncias pequenas

Na Tabela 1 são reportados os resultados para execução da busca local iterativa *multi-start* e da formulação em instâncias pequenas. A primeira coluna identifica as instâncias. As seis colunas seguintes reportam o valor da solução obtida ($valor$) e o tempo gasto ($t(s)$) por: (a) heurística *multi-start* (MSILS) sem a execução da busca em grande vizinhança, (b) busca em grande vizinhança (LNS), e (c) formulação de programação inteira (MIP). A última coluna (ótimo) identifica o valor da solução ótima. Os resultados reportados em cada linha são a média de valores para as 5 instâncias com as mesmas características (tamanho, arestas adjacentes e intervalo de pesos). Valores em negrito indicam a primeira etapa da técnica na qual a solução ótima foi encontrada.

Tabela 1: Resultados preliminares usando as heurísticas propostas e a correspondência com as soluções ótimas. Instâncias Pequenas.

Instâncias	MSILS		LNS		MIP		ótimo
	valor	t(s)	valor	t(s)	valor	t(s)	valor
$S\_R1\_25\_33\_10\_25$	**63.8**	< 1.0	63.8	< 1.0	63.8	< 1.0	63.8
$S\_R11\_50\_85\_10\_50$	281.4	2.8	**280.8**	< 1.0	280.8	< 1.0	280.8
$S\_R23\_75\_490\_10\_50$	1232.8	13.8	1232.4	57.8	**1226.6**	113.0	1226.6
$S\_SG3\_5\_5\_10\_75$	**312.4**	< 1.0	312.4	< 1.0	312.4	< 1.0	312.4
$S\_SG8\_9\_9\_10\_50$	754.8	7.4	**752.2**	1.4	752.2	1.4	752.2
$S\_NG7\_12\_6\_10\_25$	402.0	4.8	**398.2**	1.6	398.2	1.4	398.2
$S\_T2\_5\_5\_10\_50$	**124.4**	1.1	124.4	< 1.0	124.4	< 1.0	124.4
$S\_T4\_7\_7\_10\_25$	197.4	3.8	**195.4**	< 1.0	195.4	< 1.0	195.4
$S\_T7\_9\_9\_10\_25$	313.6	9.2	**309.6**	12.4	309.6	5.4	309.6
$S\_H1\_16\_10\_25$	73.8	< 1.0	**72.2**	< 1.0	72.2	< 1.0	72.2
$S\_H5\_32\_10\_50$	**240.6**	2.1	240.6	< 1.0	240.6	< 1.0	240.6
$S\_H6\_32\_10\_75$	**277.6**	2.1	277.6	< 1.0	277.6	< 1.0	277.6

Os resultados na Tabela 1 mostram que, considerando as instâncias testadas, em somente uma instância randômica as soluções ótimas não foram encontradas pela heurística. Pode-se ver que LNS trouxe uma melhoria considerável nos resultados obtidos pela etapa inicial da heurística, permitindo a obtenção de instâncias ótimas para 6 grupos adicionais de instâncias.

4.2. Instâncias grandes

Na Tabela 2 são reportados os resultados para as instâncias grandes. A estrutura desta tabela é similar à da Tabela 1, exceto pela última coluna que é substituída pela coluna *melhor*, visto que a solução ótima não é conhecida para essas instâncias. Os valores da última coluna foram retirados de Carrabs et al. [2014]. Valores em negrito indicam a primeira etapa da técnica na qual uma solução melhor ou igual à melhor conhecida foi encontrada. Um * indica que a melhor solução conhecida na literatura foi encontrada pela abordagem proposta neste trabalho.

Os resultados na Tabela 2 mostram que, novamente, LNS possibilitou uma grande melhoria nos resultados obtidos pela metaheurística *multi-start*. Cinco melhores resultados foram encontrados, considerando-se que em três casos as soluções encontradas são estritamente melhores do que as melhores conhecidas na literatura. Para dois grupos de instâncias toroidais, as soluções foram aprimoradas na etapa MIP e novos melhores resultados foram encontrados.

5. Comentários finais

Este trabalho apresentou uma nova abordagem para o problema do conjunto de vértices de retroalimentação de peso mínimo que consiste em resolver o problema dual, i.e., floresta induzida de peso máximo. Propôs-se (a) uma nova formulação compacta de programação inteira mista baseada em fluxos, (b) uma heurística de busca local iterativa *multi-start* com uma etapa de aprimoramento final baseado em uma busca local em grande vizinhança utilizando a formulação de programação inteira.

Resultados preliminares baseados nos experimentos computacionais realizados mostram que a abordagem é bastante promissora. A solução ótima foi encontrada pela heurística proposta

Tabela 2: Resultados preliminares usando o modelo proposto e a correspondência com as melhores soluções conhecidas até momento. Instâncias Grandes.

Instâncias	MSILS		LNS		MIP		melhor
	valor	t(s)	valor	t(s)	valor	t(s)	valor
$L\_R2\_100\_247\_10\_50$	837.6	13.8	**836.8**	7.0	836.8	6.0	836.8
$L\_R11\_200\_796\_10\_50$	2432.4	90.9	2432.0	60.0	2425.4	600.0	**2399.0**
$L\_R19\_300\_1644\_10\_25$	2078.0	297.0	2076.2	60.0	2076.2	600.0	**2045.4**
$L\_SG1\_10\_10\_10\_25$	571.8	9.1	**566.2***	6.4	566.2	2.6	566.8
$L\_NG3\_13\_7\_10\_75$	1382.6	7.5	**1381.6***	2.0	1381.6	1.6	1382.8
$L\_NG9\_23\_13\_10\_75$	5332.8	135.1	5325.2	60.0	5258.4	600.0	**5258.0**
$L\_T2\_10\_10\_10\_50$	459.2	14.1	**457.4***	6.4	457.4	4.8	457.6
$L\_T4\_14\_14\_10\_25$	759.6	61.3	752.8	51.4	**748.0***	447.6	748.8
$L\_T9\_17\_17\_10\_75$	1511.4	185.1	1508.2	60.0	**1497.6***	422.4	1498.6
$L\_H3\_128\_10\_75$	1163.6	32.2	**1161.6**	16.4	1161.6	15.4	1161.6
$L\_H5\_256\_10\_50$	2306.4	182.8	2305.4	60.2	2299.6	600.0	**2279.6**
$L\_H7\_512\_10\_25$	3299.2	1069.3	3299.2	60.0	3296.6	600.0	**3119.0**

para quase todas as instâncias pequenas, exceto uma. Levando-se em consideração as instâncias grandes, três novos melhores resultados foram encontrados pela heurística, e dois novos no processo de enumeração utilizando programação inteira. Os resultados também mostram a importância da busca local em grande vizinhança na melhoria das soluções obtidas pela metaheurística de busca local iterativa.

Propostas de trabalhos futuros são o aprimoraramento das técnicas de busca local, utilizando-se mecanismos de memória baseados em *Tabu Search*, bem como um algoritmo *branch-and-cut* para a melhoria dos limites fornecidos pela formulação de programação inteira. Outra possibilidade é a paralelização dos métodos propostos com o intuito de diminuir o tempo computacional para a geração de soluções com boa qualidade.

Agradecimentos: Os autores agradecem ao PIBIC/UFBA-UFBA e ao CNPQ pelo apoio financeiro.

Referências

Bafna, V., Berman, P., e Fujito, T. (1999). A 2-approximation algorithm for the undirected feedback vertex set problem. *SIAM Journal on Discrete Mathematics*, 12(3):289–297.

Bar-Yehuda, R., Geiger, D., Naor, J., e Roth, R. M. (1998a). Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and bayesian inference. *SIAM journal on computing*, 27(4):942–959.

Bar-Yehuda, R., Geiger, D., Naor, J., e Roth, R. M. (1998b). Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and bayesian inference. *SIAM journal on computing*, 27(4):942–959.

Brunetta, L., Maffioli, F., e Trubian, M. (2000). Solving the feedback vertex set problem on undirected graphs. *Discrete Applied Mathematics*, 101(1):37–51.

Carrabs, F., Cerrone, C., e Cerulli, R. (2014). A memetic algorithm for the weighted feedback vertex set problem. *Networks*, 64(4):339–356.

Carrabs, F., Cerulli, R., Gentili, M., e Parlato, G. (2004). Minimum weighted feedback vertex set on diamonds. *Electronic Notes in Discrete Mathematics*, 17:87–91.

Carrabs, F., Cerulli, R., Gentili, M., e Parlato, G. (2011). A tabu search heuristic based on k-diamonds for the weighted feedback vertex set problem. In *Network Optimization*, p. 589–602. Springer.

Festa, P., Pardalos, P. M., e Resende, M. G. (1999). Feedback set problems. In *Handbook of combinatorial optimization*, p. 209–258. Springer.

Funke, M. e Reinelt, G. (1996). A polyhedral approach to the feedback vertex set problem. In Cunningham, W., McCormick, S., e Queyranne, M., editors, *Integer Programming and Combinatorial Optimization: 5th International IPCO Conference Vancouver, British Columbia, Canada, June 3–5, 1996 Proceedings*, p. 445–459, Berlin, Heidelberg. Springer Berlin Heidelberg.

Karp, R. M. (1972). Reducibility among combinatorial problems. In *Complexity of computer computations*, p. 85–103. Springer.

Kleinberg, J. e Kumar, A. (2001). Wavelength conversion in optical networks. *Journal of algorithms*, 38(1):25–50.

Peleg, D. (1997). Local majority voting, small coalitions and controlling monopolies in graphs: A review. In *Proc. of 3rd Colloquium on Structural Information and Communication Complexity*, p. 152–169.

Peleg, D. (1998). Size bounds for dynamic monopolies. *Discrete Applied Mathematics*, 86(2-3): 263–273.

Wang, C.-C., Lloyd, E. L., e Soffa, M. L. (1985). Feedback vertex sets and cyclically reducible graphs. *Journal of the ACM (JACM)*, 32(2):296–313.