X Congresso Brasileiro de Inteligência Computacional (CBIC'2011), 8 a 11 de Novembro de 2011, Fortaleza, Ceará

GRASP COM ALGORITMOS HÍBRIDOS APLICADO AO PROBLEMA DA ÁRVORE GERADORA DE CUSTO MÍNIMO CAPACITADA EM NÍVEIS

Rennan N. Toscano[1], Lucídio A. F. Cabral [1], Marcone J. F. Souza[2], Luiz S. Ochi[3]

[1]Departamento de Informática – Universidade Federal da Paraíba (UFPB)
Caixa Postal 5125 – 58.059-900 – Paraíba – PB – Brasil

[2]Departamento de Computação – Universidade Federal de Ouro Preto (UFOP)
Ouro Preto – MG – Brazil

[3]Instituto de Computação – Universidade Federal Fluminense (UFF)
Niterói – RJ – Brazil

rennan@lavid.ufpb.br, lucidio@di.ufpb.br, marcone@iceb.ufop.br, satoru@ic.uff.br

Resumo – Este artigo propõe um algoritmo híbrido, baseado na metaheurística *Greedy Randomized Adaptive Search* (GRASP), para obter soluções melhores e reduzir o custo computacional para o problema conhecido como o Problema da Árvore Geradora de Custo Mínimo Capacitada em Níveis (PAGCMCN). Este problema, normalmente encontrado ao se projetar uma rede, consiste em determinar a melhor maneira de se conectar vários terminais a um computador central, de forma a atender suas demandas. As conexões podem envolver diferentes tipos de linhas de transmissão, onde cada tipo possui uma capacidade máxima de transmissão distinta. No algoritmo proposto, o otimizador CPLEX 12.2 é acionado, tanto na fase de construção quanto na de busca local, com vistas a uma exploração mais efetiva do espaço de busca. Os experimentos computacionais mostraram que o algoritmo proposto é bastante competitivo, superando em várias instâncias os resultados da literatura e obtendo resultados muito próximos nas demais.

Palavras-chave – Árvore geradora de custo mínimo capacitada em níveis, GRASP, Projeto de redes.

Abstract – This article proposes a hybrid algorithm, based on *Greedy Randomized Adaptive Search* (GRASP) metaheuristic, to obtain better solutions and reduce the computational cost for the problem known as the Multi-Level Capacitated Minimum Spanning Tree Problem (MLCMSTP). This problem, typically found when designing a network, is to determine the best way to connect multiple terminals to a central computer, to attend their demands. The connection may involve different types of transmission lines, where each type have distinct maximum capacities of transmission. In the proposed algorithm, the optimizer CPLEX 12.2 is triggered, both during construction and in local search phase with the aim of more effective exploitation of the search space. The computational experiments showed that the proposed algorithm is quite competitive, exceeding in many instances the results of the literature and obtaining very close results in the others.
Keywords – Multi-level capacitated minimum spanning tree, GRASP, network design.

1. INTRODUÇÃO

Com o crescimento da população e com os computadores cada vez mais acessíveis, as redes de computadores têm se tornado cada vez maiores e mais complexas, dificultando cada vez mais a busca pelas soluções de menor custo para o projeto da rede em tempo aceitável.

O presente trabalho tem como foco projetar a topologia de uma rede de forma a atender a necessidade de seus usuários com o custo mínimo das conexões. O problema consiste em encontrar a melhor maneira de conectar n terminais, espalhados geograficamente, a um computador central, de forma que eles possam enviar e receber informações do mesmo. A conexão entre um terminal e o computador central pode ser feita de forma direta, ligando os dois por uma linha de transmissão, ou de forma indireta, ligando o terminal a outro que já esteja conectado ao computador central, compartilhando, então, a(s) linha(s) de transmissão usadas pelo mesmo para comunicação com o computador central.

Modelando o problema na forma de um grafo $G = (N, A)$, sendo N o conjunto de nós formado pelo computador central e pelos terminais e A o conjunto de arestas formado pelas linhas de transmissão que conectam os nós, a topologia ótima para ele é uma árvore geradora tendo como sua raiz o computador central. Até este ponto, a solução ótima seria a solução do Problema da Árvore Geradora Mínima (PAGCM) para o grafo e poderia ser resolvido em tempo polinomial; entretanto, as linhas de transmissão não podem transmitir uma quantidade infinita de fluxo e devido à capacidade máxima Q de fluxos passando simultaneamente por elas, há um número máximo de terminais que podem utilizar essa linha para comunicar-se com o nó raiz.

Ao levar esse limite em consideração, sua complexidade aumenta significativamente e o mesmo passa a ser o que é conhecido na literatura como Problema da Árvore Geradora de Custo Mínimo Capacitada (PAGCMC), um problema NP-Completo [Amberg 1996, Sharaiha 1997, Ahuja 2001].

Por outro lado, existem vários tipos de linhas de transmissão disponíveis, com diferentes capacidades e preços. Ao permitir que as comunicações sejam feitas por mais de um tipo de linha de comunicação, reduz-se o desperdício, conseguindo-se soluções melhores, mas o número de soluções viáveis cresce exponencialmente em função da quantidade de tipos de transmissão, dificultando ainda mais a obtenção da topologia ótima. Ao adicionar esta característica, o PAGCMC passa a ser o que é conhecido como o Problema da Árvore Geradora de Custo Mínimo Capacitada em Níveis (PAGCMCN).

Para não aumentar ainda mais a complexidade, será assumido que haverá apenas uma linha de transmissão ligando dois nós (grafo não-bifurcado) e que a demanda é a mesma para todos os terminais. Para simplificar a visualização do problema, a demanda de cada terminal será unitária e as capacidades das linhas passam a ser descritas com o número de terminais que podem ser atendidos simultaneamente pelo cabo.

Na Figura 1 temos em (a) o grafo de uma instância pequena (algumas arestas do grafo estão ocultas com o intuito de facilitar o entendimento) e a distância entre os nós estão em hectômetros (100m); em (b) a solução ótima do PAGCM sobre o grafo, quando a capacidade da linha suporta tantas demandas quanto necessárias, com custo de R$ 5,00 por metro de linha usada; em (c) a solução ótima do PAGCMC, quando as linhas têm capacidade máxima para atender 3 nós, com custo de R$ 2,00 por metro de linha usado; e em (d) a solução ótima do PAGCMCN com dois tipos de linha: uma semelhante à do PAGCMC e outra com capacidade unitária e custo R$ 1,00 por metro. Sob as condições citadas, os custos das três soluções seriam, respectivamente, R$ 8.000,00, R$ 4.000,00 e R$ 3.400,00, o que mostra uma diferença significante no orçamento mesmo em uma instância bem pequena.

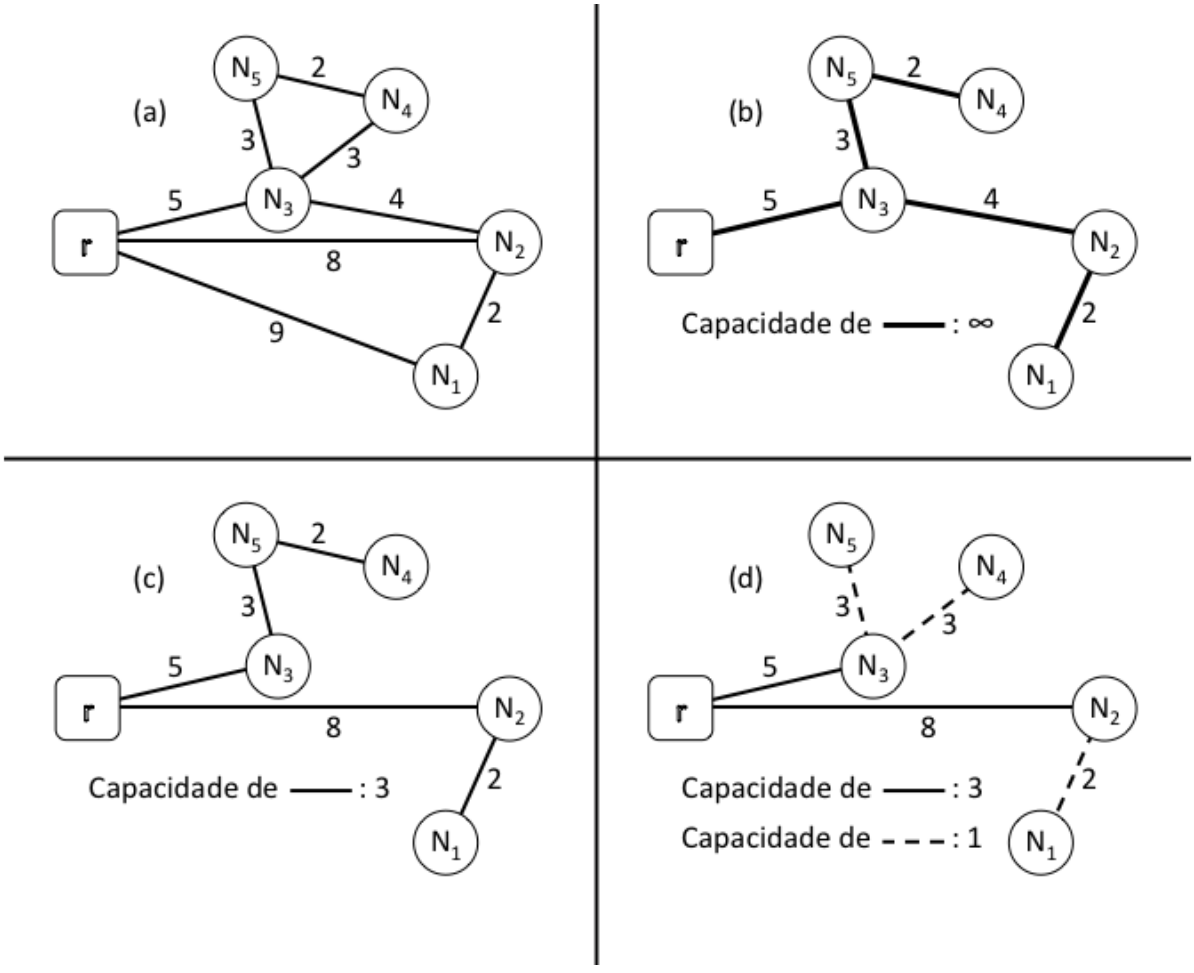

Figura 1: Instância solucionada nas três versões de complexidade do problema

Em virtude da alta complexidade, o PAGCMCN é normalmente resolvido por meio de algoritmos heurísticos de solução, como os encontrados em [Berger 2000, Gamvros 2006, Martins 2009].

Neste trabalho, propomos um algoritmo híbrido, que combina a flexibilidade de um método heurístico, com o poderio de um otimizador de mercado. O método heurístico usado para explorar o espaço de soluções do problema é o *Greedy Randomized Adaptive Search Procedure* – GRASP [Feo 1995], enquanto o otimizador é o CPLEX 12.2, versão acadêmica. No algoritmo proposto, a fase de construção GRASP consiste em agrupar e formar componentes a partir de nós próximos entre si. Já a fase de

busca local consiste em reconstruir trechos menores do grafo associado ao problema, pela aplicação do otimizador CPLEX ao modelo de programação matemática vinculado ao problema.

O restante deste trabalho está organizado como segue. Na Seção 2 temos uma breve explanação sobre o funcionamento da metaheurística GRASP. Na Seção 3 é apresentada a definição formal do problema. Na Seção 4 são mencionadas as estruturas de vizinhanças e os modelos matemáticos que foram estudados e testados no desenvolver da pesquisa. Na Seção 5 são descritos os algoritmos atualmente utilizados no experimento. Na Seção 6 são exibidos os resultados do experimento descrito. Por fim, na Seção 7 são relatadas as conclusões obtidas durante a pesquisa e são discutidos os resultados obtidos até o momento.

2. GRASP

GRASP (*Greedy Randomized Adaptive Search Procedure* - Procedimento de busca adaptativa gulosa e randômica) é um método iterativo, proposto em [Feo 1995], que consiste de duas fases: uma fase de construção, na qual uma solução é gerada, elemento a elemento, e uma fase de busca local, na qual um ótimo local na vizinhança da solução construída é pesquisado. A vizinhança de uma solução é o conjunto de soluções próximas a ela que podem ser obtidas a partir da mesma ao se executar algum movimento de transformação. A melhor solução encontrada ao longo de todas as iterações GRASP realizadas é retornada como resultado. O pseudocódigo deste procedimento para um problema de minimização é descrito pelo Algoritmo 1.

Algorithm 1: GRASP

```
Entrada: α, GRASPmax, Vizinhança N, Função de avaliação f, Função guia g
Saída: s
início
    f ← ∞;
    para Iter=1 até GRASPmax faça
        Construcao(g(.), α, s);
        BuscaLocal(g(.), α, s);
        se (f(s) < f*) então
            s* ← s;
            f* ← f(s);
        fim
    fim
    s ← s*;
    Retorne s;
fim
```

Na fase de construção, uma solução é iterativamente construída, elemento por elemento. A cada iteração desta fase, os próximos elementos candidatos a serem incluídos na solução são colocados em uma lista C de candidatos, seguindo um critério de ordenação pré-determinado. Este processo de seleção é baseado em uma função adaptativa gulosa g que estima o benefício da seleção de cada um dos elementos. A heurística é dita adaptativa porque os benefícios associados com a escolha de cada elemento são atualizados em cada iteração da fase de construção para refletir as mudanças oriundas da seleção do elemento anterior. A componente probabilística do procedimento reside no fato de que cada elemento é selecionado de forma aleatória a partir de um subconjunto restrito formado pelos melhores elementos que compõem a lista de candidatos, cujo tamanho é controlado pelo parâmetro α, que regula o nível de gulosidade e aleatoriedade do procedimento. Este subconjunto recebe o nome de lista de candidatos restrita (LCR). Esta técnica de escolha permite que diferentes soluções sejam geradas a cada iteração GRASP.

Assim como em muitas técnicas determinísticas, as soluções geradas pela fase de construção do GRASP provavelmente não são localmente ótimas com respeito à definição de vizinhança adotada. Daí a importância da fase de busca local, que tem como objetivo melhorar a qualidade da solução construída.

O procedimento GRASP procura, portanto, conjugar bons aspectos dos algoritmos puramente gulosos, com aqueles dos procedimentos aleatórios de construção de soluções.

3 DEFINIÇÃO FORMAL DO PROBLEMA

Seja um conjunto de nós $N = \{0, 1, \cdots, n\}$ e L tipos de conexões distintas, em que a cada nó $i \in N - \{\textit{nó central}\}$ está associada uma demanda positiva b_i a ser atendida pelo nó central (que não possui demanda) e a cada tipo $l \in \{1, \cdots, L\}$ está associada uma capacidade positiva z_l. O nó central, por decisão arbitrária, sempre será o nó 0 em N.

Seja $G = (N, A)$ um grafo não direcionado e conexo, sendo A um conjunto de arestas, contendo L arestas distintas para cada par de nós $(i, j) \in N^2$. Cada aresta $(l, i, j) \in A$ possui capacidade igual a z_l e custo positivo c_{ij}^l fornecido pela instância, com $l \in \{1, \ldots, L\}$. O problema consiste em encontrar uma árvore geradora de G com menor custo $C(T)$ que possua o nó central como raiz e que não viole a capacidade das arestas utilizadas (a capacidade de uma aresta é dita violada quando ela é excedida pela soma das demandas de cada nó cujo caminho até o nó raiz inclui essa aresta).

Um grafo $G' = (N', A')$ é um subgrafo de $G = (N, A)$ se $N' \subseteq N$ e $A' \subseteq A$. Uma árvore geradora de G é um subgrafo $T = (N, A')$ que é conexo e cujo A' possua exatamente $|N| - 1$ arestas. O custo $C(T)$ desta árvore é calculado como: $C(T) = \sum_{(l,i,j)\in A'} c_{ij}^l$. Caso haja apenas um tipo de aresta ($L = 1$), o problema passa a ser o PAGCMC e, nesse caso, basta apenas verificar se o somatório excedeu o limite Z_L nas arestas incidentes do nó central, uma vez que a demanda passando por qualquer aresta precisa passar por uma delas antes de alcançar o nó raiz.

Quando a demanda é homogênea, ou seja, todos os nós exceto a raiz possuem demandas iguais ($\forall_{i \in N-\{0\}}, b_i = B$), o problema se reduz a encontrar uma árvore geradora na qual cada uma das arestas possuem no máximo $\lfloor z_l/B \rfloor$ nós utilizando-a. Portanto, qualquer instância com demanda homogênea pode ser convertida para uma instância com demanda unitária e capacidades $\{\lfloor z_1/B \rfloor, \lfloor z_2/B \rfloor, ..., \lfloor z_L/B \rfloor\}$.

Para uma instância do PAGMC com $Z_L = 1$, a melhor e única solução é conectar todos os nós diretamente ao nó central. Para $Z_L = 2$, [Hall 1996] mostra que o problema pode ser resolvido como um problema de emparelhamento (*weighted matching problem*). Para $Z_L \geq n$, a restrição é irrelevante e basta encontrar uma árvore geradora mínima não capacitada. [Papadimitriou 1978] mostrou que o PAGCMC é NP-difícil para $3 \leq Z_L \leq \lfloor |N|/2 \rfloor$ e, portanto, não se conhecem algoritmos exatos que resolvam em tempo polinomial o PAGCMC sob estas condições e, em se tratando de uma generalização do PAGCMC, também não se conhecem para o PAGCMCN.

4 REVISÃO DA LITERATURA

Apesar de ser uma versão generalizada do PAGMC, o PAGMCN não obteve tanta atenção dos pesquisadores quanto o PAGMC. Esse fato é surpreendente, haja vista que o PAGMCN permite o uso de mais de um tipo de linhas de transmissão entre os pontos na rede e, por isso, se aproxima mais da realidade e ainda possibilita gerar soluções de menor custo. Das poucas abordagens encontradas para o problema, citamos: [Berger 2000], [Gamvros 2002], [Gamvros 2006] e [Martins 2009]. Em face à estrutura semelhante, foram utilizadas nesses trabalhos apenas estruturas de vizinhanças apresentadas anteriormente em trabalhos sobre o PAGMC.

4.1 Estruturas de vizinhança

A vizinhança proposta em [Amberg 1996] é composta por dois tipos de movimentos a partir de uma solução viável: *node shift* e *node exchange*. O movimento *node shift* transfere um nó de uma componente (sub-árvore incidente à raiz), coloca-o em outra componente que não esteja no limite e então reconstrói as duas componentes mantendo seus nós. Já o movimento *node exchange* consiste em escolher dois nós em duas componentes diferentes, trocá-los e reconstruir as duas componentes.

A vizinhança proposta em [Sharaiha 1997] é constituída pelo movimento *cut and paste*, onde basicamente uma sub-árvore é cortada de dentro de uma componente e então colada na raiz ou em um nó de outra componente que possa comportar todos os nós da sub-árvore sem violar a restrição de capacidade.

4.2 Modelos matemáticos

Foram encontrados na literatura três formulações matemáticas diferentes para o problema. A primeira, apresentada em [Gamvros 2002], é baseada em fluxos, possui três tipos de variáveis e, diferentemente das outras formulações para PAGCMC e PAGCMCN, possui seu fluxo partindo dos nós em direção à raiz. A segunda, apresentada em [Martins 2005], é uma generalização do modelo baseado em fluxos para a PAGCMC, apresentado em [Gavish1982]. A terceira, também apresentada em [Martins 2005], é uma modificação da segunda com o intuito de reduzir o número de variáveis em detrimento ao tamanho das restrições. Dentre elas, a que apresentou um desempenho bem superior as demais foi a terceira, conhecida como Modelo Baseado na Capacidade (MBC), e foi incorporada ao algoritmo proposto. Sua formulação é apresentada pelas equações (4.1) a (4.4).

$$\textit{Minimize} \sum_{l=1}^{L} \sum_{i=0}^{n} \sum_{j=1}^{n} c_{ij}^{l} x_{ij}^{l} \tag{4.1}$$

$$\text{Sujeito a} \sum_{l=1}^{L} \sum_{i=0}^{n} x_{ij}^{l} = 1 \qquad j = 1, \ldots, n \tag{4.2}$$

$$\sum_{l=1}^{L} \sum_{i=0}^{n} z_l x_{ij}^{l} - \sum_{l=1}^{L} \sum_{i=1}^{n} z_l x_{ji}^{l} \geq b_j \qquad j = 1, \ldots, n \tag{4.3}$$

$$x_{ij}^{l} \in \{0, 1\} \tag{4.4}$$

Nessa formulação, x_{ij}^{l} é igual a 1 se os nós i e j são conectados por uma linha do tipo l e igual a 0, caso contrário; c_{ij}^{l} é o custo de conectar os nós i e j utilizando uma linha do tipo l; z_l é a capacidade máxima de uma linha do tipo l e b_j é a demanda do nó j. As restrições 4.2 garantem que a topologia será uma árvore ao obrigar todos os nós a terem exatamente um pai, exceto a raiz que não possui pai, enquanto que as restrições 4.3 garantem que o fluxo não exceda a capacidade da linha e que a demanda b_j seja atendida.

5. SOLUÇÃO PROPOSTA

A solução proposta, por ser baseada na metaheurística GRASP, é dividida em duas fases principais: a fase de construção e a fase de busca local, que são descritas a seguir. Em virtude do tempo gasto em instâncias maiores, decidiu-se aplicar apenas 6 iterações do algoritmo. Em cada iteração a fase de construção é executada 5 vezes e a melhor construção é entregue à busca local, que continua o processo iterativo enquanto houver melhora. As soluções entregues pela busca local são comparadas e a melhor dentre todas é apresentada. As subseções seguintes descrevem cada uma dessas fases.

5.1 Fase de construção

A fase de construção baseia-se na formação de componentes (sub-árvores incidentes ao nó central) em torno de nós aleatórios. Primeiramente, colocamos o nó central na solução como uma árvore unitária. Então selecionamos um nó fora da árvore aleatoriamente e aplicamos o modelo matemático nele junto com a raiz e os Q nós mais próximos que estejam fora da árvore, para formar a(s) melhor(es) componente(s) possíveis envolvendo apenas esses nós. Um nó j é dito próximo do nó i quando a distância entre eles é no máximo 20% maior que a distância entre i e o nó mais próximo de i. A quantidade de nós Q que integrarão o modelo junto com o nó escolhido é arbitrária. Neste trabalho, essa quantidade equivale a duas vezes o número máximo de nós que uma linha de transmissão pode suportar menos 1. Mesmo que não existam Q nós próximos do nó selecionado, o modelo é aplicado apenas com os nós dentro desse limite de 20%. O processo de seleção e aplicação do modelo é repetido até que todos os nós estejam na árvore.

O algoritmo procura manter nós próximos dentro da mesma componente ou em componentes vizinhas, uma vez que as chances de dois nós distantes estarem conectados na solução ótima é mínima (ou zero se a distância entre eles for maior que a distância para o nó central), permitindo, então, que a busca local se concentre em movimentos entre componentes próximas. A desvantagem é que quando restarem poucos nós fora da árvore, haverão dentre eles poucos nós próximos ao selecionado gerando, portanto, algumas pequenas componentes que levarão alguns movimentos na busca local até se unirem a componentes maiores.

5.2 Fase de busca local

Foram feitos testes e adaptações nas estruturas de vizinhanças descritas na seção 4.1, principalmente com o movimento *node shift*. Entretanto, ainda não foram mesclados ao algoritmo mais recente de busca local de forma a trazer melhoras significativas e, como não justificavam seus custos computacionais, foram temporariamente removidos das últimas versões geradas, incluindo a versão que gerou os resultados apresentados neste trabalho.

A estratégia para o aprimoramento da solução construída consiste na reconstrução de trechos menores do grafo por meio do modelo matemático. Devido ao custo das arestas serem derivados das distâncias geográficas, é fácil notar que à medida que a solução atual se aproxima da solução ótima, os vértices vão ficando na mesma componente ou em componentes vizinhas das componentes dos vértices que lhe são mais próximos. Outro fato observado é que se aplicarmos a reconstrução de forma ótima em todos os nós (e apenas neles) que componham algumas componentes da solução ótima obtemos exatamente essas componentes. Tomando essas observações como base, a estratégia tem como foco aplicar a reconstrução de componentes próximas umas das outras, para que os vértices internos a elas alcancem, ou ao menos se aproximem, da componente em que deveriam ficar.

O algoritmo inicia calculando os ângulos entre os nós e a raiz, e a partir deles, ordena as componentes em sentido anti-horário. São selecionados diferentes grupos de componentes, aplicando-se a reconstrução no grupo inteiro. Um grupo inicia com uma componente escolhida aleatoriamente, sendo adicionadas as componentes próximas no sentido anti-horário até que o limite de número de nós ou o tamanho máximo aceitado para o arco que contém o grupo seja atingido. Uma parte dos grupos é formada adicionando a componente vizinha da que foi adicionada mais recentemente e outra parte é formada desprezando as vizinhas e adicionando as que vêm depois delas. Essa segunda formação tem como objetivo acelerar o deslocamento de um nó que esteja distante de seus próximos e também evitar casos onde um nó não consegue se transferir para outra componente por haver muitos nós intermediários e a reconstrução não abranger todos os nós da sua componente ideal.

O limite de nós dentro de um grupo inicia-se com 30 e aumenta à medida que nenhuma melhora é obtida. Esse limite vai aumentando até atingir 50, devido ao custo computacional sobre o modelo matemático. O tamanho máximo do arco permite evitar desperdício de esforço computacional, mas afeta levemente a qualidade da solução. Como o tempo de execução estava relativamente baixo, foi utilizado na versão que gerou os resultados um limite de 360º, que serve apenas para evitar repetição de componentes no grupo.

6. RESULTADOS

Nesta seção relataremos resultados numéricos obtidos em um subconjunto de instâncias de referência introduzida na literatura em [Gamvros 2006]. Cada instância contêm um grafo com 50, 100 ou 150 nós além do nó central distribuídos em um *grid* quadrado 40×40. Os nós possuem demanda unitária e as instâncias são divididas em três categorias de acordo com a posição no grafo do nó central (localizado no centro, na ponta ou em uma posição aleatória). Existem 3 tipos de linhas de transmissão possuem capacidade das linhas são 1, 3 e 10, e os custos são $1d_{ij}$, $2d_{ij}$ e $6d_{ij}$ respectivamente, onde d_{ij} é a distância Euclidiana (não arredondada).

A aplicação foi codificada na linguagem C++, e compilada usando o compilador gcc 4.2.1 e o CPLEX 12.2, versão Acadêmica. As execuções das instâncias apresentadas nesta seção foram feitas em um notebook Intel Core 2 Duo 2.26Ghz com 4 GB de RAM rodando o sistema operacional Mac OS X 10.6.7. Devido à estratégia utilizada, a execução em instâncias com 50 nós ou menos levará sem dúvida à execução do modelo matemático sobre todo o grafo e obterá, assim, a solução ótima. Dado o fato de que esses resultados podem ser alcançados com o modelo de **??**, não os exibiremos. Para os demais casos, os melhores resultados até o momento pertencem a [Uchoa 2011]. Esses autores adicionaram estratégias de *branch-and-cut* ao algoritmo híbrido proposto em [Martins 2009], conseguindo, com isso, resultados bem mais próximos aos *lower bounds* (limite inferior - maior valor encontrado que comprovadamente não pode ser obtido por uma solução viável) em todas as instâncias.

Em [Uchoa 2011], os autores propuseram dois experimentos com o algoritmo proposto. O primeiro (Uchoa 1) teve como foco obter uma solução aceitável em um curto período de tempo e produziu soluções com gaps de até 1,25% em todas as instâncias

e 1,14% contando apenas as instâncias mencionadas nas tabelas. O segundo (Uchoa 2) teve como foco a qualidade da solução, conseguindo *gaps* de, no máximo, 0,88% em todas as instâncias e 0,80% nas instâncias mencionadas nas tabelas. Entretanto, as execuções levaram até 6h para obter a solução de uma instância. O experimento com o algoritmo proposto foi feito também visando obter uma solução em um curto período de tempo e obteve soluções com *gaps* de até 0,90% nas instâncias mencionadas nas tabelas.

Na Tabela 1 temos o resultado do algoritmo proposto em instâncias com solução ótima conhecida. Em [Uchoa 2011] foi apenas dito que o primeiro experimento obteve 17 ótimos dentre as 50, enquanto que o segundo obteve 47 e obtiveram *gap* médio de 0,14% e 0,005% respectivamente e, portanto, seus resultados não estão incluídos na tabela.

Tabela 1: Resultados obtidos para grafo com 100 nós com o nó central localizado no centro

Instância	Ótimo	GRASP	tempo(s)	gap
0	1075,429	1075,429	713	0,00%
1	1102,352	1102,352	710	0,00%
2	1108,356	1108,356	727	0,00%
3	1096,077	1096,077	566	0,00%
4	1073,83	1073,980	778	0,01%
5	1106,464	1106,464	558	0,00%
6	1042,484	1042,656	811	0,02%
7	1136,44	1136,440	807	0,00%
8	1104,339	1104,339	684	0,00%
9	1140,361	1140,361	562	0,00%

Nas tabelas 2 a 6 vemos um comparativo entre os três experimentos. As tabelas contêm, respectivamente: o índice da instância, o maior *lower bound* (LB) conhecido e, para cada um dos experimentos, é mostrado o custo da melhor solução do experimento seguido do tempo gasto e o gap em relação ao *lower bound*. Na última coluna, classificamos a qualidade da solução de 1 a 5 estrelas baseado na comparação com os experimentos de uchoa. 1 estrela ($\star$) indica que a solução foi inferior aos dois experimentos, 2 estrelas ($\star\,\star$) indica que a solução empatou com o primeiro, 3 estrelas ($\star\,\star\,\star$) indica que a solução superou o primeiro mas foi inferior ao segundo, 4 estrelas ($\star\,\star\,\star\,\star$) indica que a solução empatou com o segundo e 5 estrelas ($\star\,\star\,\star\,\star\,\star$) indica que a solução superou os dois experimentos.

Tabela 2: Resultados obtidos para grafo com 100 nós com o nó central localizado na ponta

Inst	LB	Uchoa 1	t(s)	gap	Uchoa 2	t(s)	gap	GRASP	t(s)	gap	Qualidade
0	2158,87	2170,79	856	0,55%	2168,10	6414	0,43%	2168,10	1854	0,43%	$\star\star\star\star$
1	2000,41	2018,45	709	0,90%	2007,63	5896	0,36%	2009,40	2102	0,45%	$\star\star\star$
2	2103,56	2113,19	533	0,46%	2111,53	5258	0,38%	2111,53	2068	0,38%	$\star\star\star\star$
3	1978,62	1991,74	678	0,66%	1989,07	5827	0,53%	1989,33	2056	0,54%	$\star\star\star$
4	2020,18	2038,51	700	0,91%	2032,88	5827	0,63%	2034,42	2376	0,70%	$\star\star\star$
5	2127,46	2139,10	342	0,55%	2136,93	4873	0,45%	2137,75	1818	0,48%	$\star\star\star$
6	2019,39	2029,86	760	0,52%	2028,61	6844	0,46%	2029,65	1971	0,51%	$\star\star\star$
7	2201,38	2210,14	576	0,40%	2209,25	5277	0,36%	2209,41	1624	0,36%	$\star\star\star$
8	2018,46	2029,32	812	0,54%	2029,32	7059	0,54%	2031,21	2020	0,63%	$\star$
9	2012,16	2028,30	461	0,80%	2021,45	4820	0,46%	2021,77	2267	0,48%	$\star\star\star$

7. CONCLUSÃO

Devido a estrutura do problema, onde ao mover um único nó requer conferir a situação de várias outras arestas e possivelmente alterá-la e que tal movimento leva ao recálculo do custo de ambas as sub-árvores envolvidas, o uso de algoritmos híbridos mostra resultados melhores e um custo computacional não muito superior aos de heurísticas puras. Diante de estratégias gulosas, que não permitem que a solução piore, prendendo-a em um ótimo local, estruturas *multi-start* como o GRASP são bastantes efetivas. O algoritmo GRASP proposto neste artigo mostrou-se bastante competitivo, obtendo as melhores soluções em 7 instâncias, empatando em 5 com o melhor da literatura dentre as 50 instâncias testadas e com *gaps* em média apenas 0,08% acima dos resultados da literatura, mesmo tendo-se como foco o menor tempo de processamento.

No futuro serão realizados experimentos mais focados na qualidade da solução que no tempo. A meta principal é reduzir o custo computacional sem que haja perda da qualidade das soluções. A ideia é focar principalmente na busca local, com um uso mais eficiente das aplicações do modelo matemático, de forma a gerar soluções de boa qualidade em poucos minutos e soluções ainda melhores caso o tempo não seja um problema.

Tabela 3: Resultados obtidos para grafo com 100 nós com o nó central localizado em uma posição aleatória

Inst	LB	Uchoa 1	t(s)	gap	Uchoa 2	t(s)	gap	GRASP	t(s)	gap	Qualidade
0	1588,09	1594,36	363	0,39%	1594,36	5954	0,39%	1596,35	1432	0,52%	⋆
1	1875,21	1885,38	654	0,54%	1885,06	6572	0,53%	1885,06	2564	0,53%	⋆⋆⋆⋆
2	1473,08	1482,11	456	0,61%	1476,98	5196	0,26%	1476,98	1654	0,26%	⋆⋆⋆⋆
3	1159,71	1168,11	434	0,72%	1166,97	3808	0,63%	1168,47	871	0,76%	⋆
4	1149,27	1158,40	396	0,79%	1155,71	3347	0,56%	1157,11	670	0,68%	⋆⋆⋆
5	1197,87	1207,35	236	0,79%	1202,53	3796	0,39%	1204,63	709	0,56%	⋆⋆⋆
6	1594,86	1603,82	797	0,56%	1602,12	5431	0,45%	1603,85	1230	0,56%	⋆
7	1100,72	1109,79	422	0,82%	1107,75	3815	0,64%	1108,68	1146	0,72%	⋆⋆⋆
8	1321,46	1336,49	787	1,14%	1326,98	4679	0,42%	1326,98	2076	0,42%	⋆⋆⋆⋆
9	1484,94	1497,35	462	0,84%	1494,31	5124	0,63%	1496,40	2954	0,77%	⋆⋆⋆

Tabela 4: Resultados obtidos para grafo com 150 nós com o nó central localizado no centro

Inst	LB	Uchoa 1	t(s)	gap	Uchoa 2	t(s)	gap	GRASP	t(s)	gap	Qualidade
0	1541,393	1551,20	696	0,64%	1550,38	7614	0,58%	1553,16	1335	0,76%	⋆
1	1627,047	1638,97	561	0,73%	1634,42	5588	0,45%	1637,26	970	0,63%	⋆⋆⋆
2	1611,212	1624,90	675	0,85%	1620,86	5681	0,60%	1623,85	1165	0,78%	⋆⋆⋆
3	1569,488	1578,88	552	0,60%	1578,88	5997	0,60%	1581,49	1186	0,76%	⋆
4	1617,757	1630,15	560	0,77%	1626,31	5594	0,53%	1627,89	1013	0,63%	⋆⋆⋆
5	1643,616	1656,97	566	0,81%	1655,03	5551	0,69%	1653,69	1229	0,61%	⋆⋆⋆⋆⋆
6	1542,732	1553,34	528	0,69%	1549,00	5437	0,41%	1553,26	932	0,68%	⋆⋆⋆
7	1583,2	1595,32	694	0,77%	1593,13	7168	0,63%	1594,61	1516	0,72%	⋆⋆⋆
8	1576,648	1586,31	471	0,61%	1584,94	6203	0,53%	1587,37	1068	0,68%	⋆
9	1536,034	1548,26	680	0,80%	1548,27	6690	0,80%	1549,61	1365	0,88%	⋆

Tabela 5: Resultados obtidos para grafo com 150 nós com o nó central localizado na ponta

Inst	LB	Uchoa 1	t(s)	gap	Uchoa 2	t(s)	gap	GRASP	t(s)	gap	Qualidade
0	2992,415	3003,72	711	0,38%	3003,72	8679	0,38%	3008,39	1984	0,53%	⋆
1	3082,438	3102,86	829	0,66%	3099,18	9653	0,54%	3101,80	1919	0,63%	⋆⋆⋆
2	3028,971	3049,30	754	0,67%	3043,19	9466	0,47%	3046,77	1450	0,59%	⋆⋆⋆
3	3008,361	3028,94	906	0,68%	3025,92	9005	0,58%	3026,23	2191	0,59%	⋆⋆⋆
4	2993,525	3013,03	735	0,65%	3012,15	8933	0,62%	3010,88	1751	0,58%	⋆⋆⋆⋆⋆
5	3080,425	3105,76	1066	0,82%	3096,33	9153	0,52%	3094,82	1809	0,47%	⋆⋆⋆⋆⋆
6	3083,835	3104,05	1246	0,66%	3104,05	10648	0,66%	3100,08	2588	0,53%	⋆⋆⋆⋆⋆
7	3080,592	3102,89	950	0,72%	3101,61	8463	0,68%	3105,48	2974	0,81%	⋆
8	2968,513	2993,71	1112	0,85%	2987,79	9761	0,65%	2983,12	1837	0,49%	⋆⋆⋆⋆⋆
9	2981,108	2998,24	1120	0,57%	2998,24	10078	0,57%	2996,80	2520	0,53%	⋆⋆⋆⋆⋆

Tabela 6: Resultados obtidos para grafo com 150 nós com o nó central localizado em uma posição aleatória

Inst	LB	Uchoa 1	t(s)	gap	Uchoa 2	t(s)	gap	GRASP	t(s)	gap	Qualidade
0	2363, 65	2384, 15	1199	0, 87%	2374, 98	9565	0, 48%	2378, 71	1546	0, 64%	⋆⋆⋆
1	2076, 39	2094, 01	680	0, 85%	2087, 43	7762	0, 53%	2092, 39	1322	0, 77%	⋆⋆⋆
2	2201, 89	2217, 36	881	0, 70%	2216, 42	8108	0, 66%	2218, 04	1614	0, 73%	⋆
3	2135, 2	2157, 34	778	1, 04%	2147, 62	7595	0, 58%	2154, 42	1332	0, 90%	⋆⋆⋆
4	2195, 59	2206, 69	552	0, 51%	2206, 68	8411	0, 51%	2213, 18	1287	0, 80%	⋆
5	2584, 25	2605, 67	1008	0, 83%	2601, 70	8974	0, 68%	2604, 67	1798	0, 79%	⋆⋆⋆
6	2421, 91	2443, 17	1253	0, 88%	2438, 75	7632	0, 70%	2442, 47	2210	0, 85%	⋆⋆⋆
7	2001, 65	2018, 55	566	0, 84%	2011, 80	6235	0, 51%	2017, 26	1188	0, 78%	⋆⋆⋆
8	2433, 12	2454, 91	1244	0, 90%	2452, 28	10030	0, 79%	2452, 03	1302	0, 78%	⋆⋆⋆⋆⋆
9	2003, 48	2016, 09	714	0, 63%	2011, 40	6934	0, 40%	2014, 90	1117	0, 57%	⋆⋆⋆

AGRADECIMENTOS

Agradecemos especialmente à CAPES, que financia esta pesquisa, ao CNPq e a Alexandre X. Martins, que nos forneceu as instâncias utilizadas neste trabalho.

REFERÊNCIAS

[Ahuja 2001] Ahuja, R.K. Orlin, J.B. e Sharma, D. (2001). Multi-exchane neighborhood structures for the capacitated minimum spanning tree problem. *Mathematical Programming*, 91:71–97.

[Amberg 1996] Amberg, A. Domschke, W. e Voss, S. (1996). Capacited minimum spanning trees: Algorithms using intelligent search. *Combinatorial Optimization: Theory and Practice*, 1:9-40.

[Berger 2000] Berger, D. Gendron, B. Potvin, J.-Y. Raghavan, S. Soriano, P. (2000). Tabu Search for a Network Loading Problem with Multiple Facilities. *Journal of Heuristics*, 6:253-267.

[Feo 1995] Feo, T. and Resende, M. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133.

[Gamvros 2002] Gamvros, I. Raghavan, S. e Golden, B. (2002). An Evolutionary Approach to the Multi-Level Capacitated Minimum Spanning Tree Problem. *Center for Satellite and Hybrid Communication Networks*, 1-22.

[Gamvros 2006] Gamvros, I. Raghavan, S. e Golden, B. (2006). The multi-level capacitated minimum spanning tree. *INFORMS Journal on Computing*, 348-365.

[Gavish1982] Gavish, B. (1982). Topological design of centralized computer networks: Formulations and algorithms. *Networks*, 12, 355-377.

[Hall 1996] Hall, L. (1996). Experience with a cutting plane approach for the capacitated spanning tree problem. *INFORMS J. Comput.*, 8,219-234.

[Papadimitriou 1978] Papadimitriou, C. (1978). The complexity of the capacited tree problem. *Networks*, 8, 217-230.

[Martins 2005] Martins, A.X., Souza, M.J.F., de Souza, M.C. (2005). Modelos matemáticos para o problema da árvore geradora mínima capacitada em níveis. *Anais do XXXVII Simpósio Brasileiro de Pesquisa Operacional*, 1971-1982.

[Martins 2009] Martins, A. X. Souza, M.C. Souza, M. J. Toffolo, T. A. (2009). GRASP with hybrid heuristic-subproblem optimization for the multi-level capacitated minimum spanning tree problem. *Journal of Heuristics, Volume 15, Issue 2*, 133-151.

[Sharaiha 1997] Sharaiha, Y.M. Gendreau, M. Laporte, G. e Osman, I.H. (1997). A tabu search al- gorithm for the Capacitated Shortest Spanning Tree Problem. *Networks*, 29,161-171.

[Uchoa 2011] Uchoa, E. Toffolo, T. A. Souza, M.C. Martins, A. X. Fukasawa, R. (2011). Branch-and-cut and hybrid local search for the multi-level capacitated minimum spanning tree problem. *Networks (New York, N.Y. Print)*.