UMA METAHEURÍSTICA GRASP PARA O PROBLEMA DA ÁRVORE GERADORA DE CUSTO MÍNIMO COM GRUPAMENTOS UTILIZANDO GRAFOS FUZZY

FABIANO VIEIRA DE ALVARENGA
MARCELO LISBOA ROCHA

Departamento de Ciência da Computação – Fundação UNIRG
Alameda Madrid Nº 545, Jardim Sevilha, CEP 77410-470, Gurupi – TO – Brasil
{fabiano_unirg, marcelolisboarocha}@yahoo.com.br

Resumo: As metaheurísticas vêm tendo um ótimo desempenho nos últimos anos quando se diz a respeito a problemas de otimização. Entre estas, a metaheurísticas GRASP (*Greedy Randomized Adaptive Search Procedure* - Procedimento de Busca Gulosa Adaptativa Aleatória) atualmente possui um grande destaque na literatura para estes tipos de problemas, isso devido aos bons resultados obtidos. Neste trabalho, a mesma será aplicada a um problema variante da Árvore Geradora Mínima (AGM) denominada Árvore Geradora de custo Mínimo com Grupamentos (AGMG). Porém, esta aplicação será com a utilização de grafos fuzzy, cujas arestas apresentam graus de pertinência. Este grau de pertinência da árvore geradora de custo mínimo com grupamentos também é interpretado como um valor de confiabilidade da solução. Assim sendo, este trabalho tem como objetivo a aplicação da metaheurística GRASP ao problema da Árvore Geradora de custo Mínimo com Grupamentos sobre grafos fuzzy, gerando assim, um conjunto de soluções fuzzy para cada instância de teste utilizada.

Palavras chaves: Metaheurística GRASP, Árvore Geradora de custo Mínimo com Grupamentos, Grafos Fuzzy.

A GRASP METAHEURISTIC TO CLUSTERED MINIMAL SPANNING TREE PROBLEM USING FUZZY GRAPHS

Abstract: Metaheuristics has been a great performance in last years when applied to optimization problems. Among them, nowdays GRASP (*Greedy Randomized Adaptive Search Procedure*) metaheuristic has a great focus in specialized literature on this kind of problems, mainly due to good results gotten. In these work, the same will be used to a problem derived from Minimum Spanning Tree (MST) problem called Minimum Spanning Tree with Clusters. However, this application will use on fuzzy graphs, whose edges present degrees of pertinence. This degree of pertinence is interpreted as a value of reliability of the solution. Thus, this work has as goal the use of GRASP metaheuristic to the Minimum Spanning Tree with Clusters problem over fuzzy graphs, generating then a fuzzy solution set to each test instance used.

Keywords: GRASP Metaheuristic, Minimum Spanning Tree with Clusters, Fuzzy Graphs.

(Received October 19, 2005 / Accepted January 24, 2006)

1. Introdução

A utilização de técnicas heurísticas visando à solução de problemas, tais como o problema da Árvore Geradora de custo Mínimo com Grupamentos utilizando Grafos Fuzzy (AGMG-GF), é justificada pela complexidade do problema em seu caso geral, o que inviabiliza a utilização de técnicas exatas para resolvê-lo.

Somente a partir dos anos 80 [7], começaram a surgir na literatura o que se denomina de metaheurísticas

ou heurísticas inteligentes, tais como: algoritmos genéticos, GRASP, busca tabu, entre outros.

As metaheurísticas [5, 6], nada mais são do que um processo iterativo ou de refinamento de solução de problema que organiza e direciona heurísticas subordinadas, pela combinação de diferentes conceitos, podendo manipular uma solução completa, incompleta ou um conjunto de soluções, tentando evitar a parada prematura em ótimo local, através de mecanismos que permitam escapar de um ótimo local, podendo assim obter um desempenho melhor que as heurísticas tradicionais.

A metaheurística GRASP [4], atualmente possui um grande destaque na literatura para problemas de otimização. Tal destaque se deve aos bons resultados obtidos nas aplicações já realizadas, pela facilidade de paralelização e principalmente por sua flexibilidade, daí o motivo da escolha desta metaheurística para ser aplicada ao problema da AGMG-GF.

Normalmente, problemas reais tais como: sistemas de irrigação, projetos de rodovias, redes de telecomunicações, computadores, telefonia, entre outros [1, 2, 8, 9, 10, 11]; têm associados uma série de parâmetros como custo, capacidades, demandas, que não são naturalmente precisos [17, 18]. Por outro lado, podem existir casos em que nem a estrutura de grafos em si é precisa. Este fato faz com que uma modelagem sob o ponto de vista da teoria de sistemas nebulosos seja muito atraente, como a utilização de grafos fuzzy .

O problema da Árvore Geradora de custo Mínimo com Grupamentos utilizando Grafos Fuzzy (AGMG-GF), consiste em um problema de grafos, que possui graus de pertinência em suas arestas. Um grafo fuzzy pode apresentar graus de pertinência ou incertezas tanto na estrutura (nós e/ou arcos), quanto nos parâmetros (custo, capacidade, demanda). Este grau de pertinência também pode ser interpretado como um valor de confiabilidade da solução.

2. Revisão da Literatura

Nesta Seção é apresentada uma breve descrição do problema da Árvore Geradora Mínima (AGM) para um melhor entendimento sobre o problema da Árvore Geradora de custo Mínimo com Grupamentos (AGMG) que é descrito logo em seguida. Após estas descrições é feita uma abordagem sobre Grafos Fuzzy e, posteriormente é mostrada a definição do problema da Árvore Geradora de custo Mínimo com Grupamentos utilizando Grafos Fuzzy (AGMG - GF).

2.1. Árvore Geradora Mínima

Uma árvore é um grafo conexo sem ciclo. Dado um grafo conexo $G=(V, E)$, uma árvore T é denominada árvore geradora de G, se $T=(V', E')$ é um subgrafo conexo e sem ciclos contendo todos os vértices de G, ou seja, $V=V'$ e $E' \subseteq E$. Por exemplo, na Figura 1, o subgrafo indicado em linhas grossas é uma árvore geradora possível do grafo representado. Vale a pena ressaltar que uma árvore geradora com $|V'|=n$ vértices possui $|E'|=n-1$ arestas.

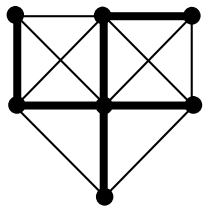

Figura 1: Grafo Conexo.

Dado um grafo conexo $G=(V, E)$ com custos (pesos) associado a cada aresta, uma árvore T é denominada Árvore Geradora Mínima (AGM) de G se T é um subgrafo de G contendo todos os vértices de G, ou seja, se for árvore geradora, e possuir o menor (mínimo) custo total (soma dos custos das arestas), dentre todas as árvores geradoras possíveis. A AGM do grafo da Figura 1 está representada na Figura 2.

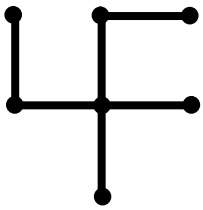

Figura 2: Árvore Geradora Mínima.

2.2. Árvore Geradora de Custo Mínimo com Grupamentos

O problema da Árvore Geradora de custo Mínimo com Grupamento (AGMG) se dá através de um conjunto de vértices do grafo associado particionado em subconjuntos disjuntos, onde a árvore geradora resultante não deve ligar necessariamente todos os vértices e sim todos os conjuntos.

O problema da variante da Árvore Geradora Mínima (AGM), aqui denominada Árvore Geradora de Custo Mínimo com Grupamento (AGMG), é muito pouco explorado na literatura. O trabalho mais relevante foi o apresentado em [3], onde são apresentadas uma formulação matemática, três heurísticas de construção e uma de busca local, além de um Algoritmo Genético

(AG). A segunda referência é o trabalho apresentado em [7], onde os autores apresentam um Algoritmo Genético (AG) incorporando módulos de busca local.

A AGMG consiste em um grafo $G = (V, E)$ onde V representa um conjunto de vértices e E é o conjunto de arestas que ligam os vértices de V. O conjunto de vértices V é particionado em k grupamentos, onde $V = V_1 \cup V_2 \cup ... \cup V_k$. O objetivo da AGMG é ligar pelo menos um nó (vértice) de cada grupamento (conjunto), de modo que a árvore geradora possa ligar todos os grupamentos do grafo com custo mínimo, onde custo significa a soma dos valores de todas as arestas que formam a AGMG.

Esta variante da AGM é classificada como NP-Difícil [3]. Somente para alguns casos particulares, este problema possui complexidade polinomial [7]. Por exemplo, quando o número de grupamentos for igual a um, a solução é formada por um único vértice. Já, quando houver exatamente dois grupos, a solução será uma aresta de menor peso conectando estes dois grupos.

Um outro caso particular simples de ser resolvido é quando cada grupamento possui um único elemento, reduzindo a AGMG ao problema da AGM [7].

Na Figura 3, tem-se um exemplo de uma aplicação para sistemas de irrigação, onde se tem uma área (representada por um grafo) particionada em 8 regiões (grupamentos). O problema consiste em criar uma rede de irrigação de menor caminho que possa tocar no mínimo um vértice de cada região, onde o nó fonte é o principal, pois é nele que se localiza a fonte de água. Esta rede não pode cruzar as regiões, podendo apenas passar pelas suas fronteiras (arestas). Assim, este problema da irrigação pode ser modelado com uma AGMG. Considerando um Grafo $G=(V, E)$, os vértices de V representam os vértices das fronteiras das regiões, incluso a fonte e, o peso de cada aresta de E é associado à distância entre os seus vértices.

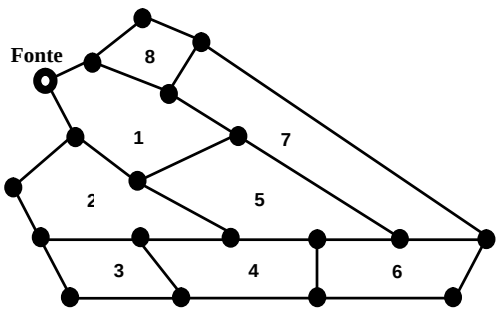

Figura 3: Uma aplicação para o problema da AGMG.

A Figura 4 apresenta uma solução para o sistema de irrigação da Figura 3, onde se observa a Árvore Geradora de Custo Mínimo com Grupamentos. Na Figura 4, AGMG está representada por linhas mais grossas, partindo do nó fonte e percorrendo todos os grupamentos (regiões) do grafo, ligando pelo menos um vértice de cada grupamento, sem cruzar as fronteiras (arestas) ou formar ciclos.

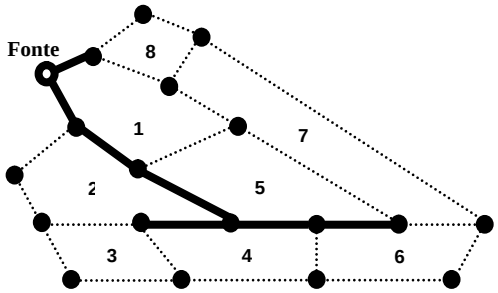

Figura 4: Solução para a Figura 3.

2.3. Grafos Fuzzy

Em [17, 18] são mostrados alguns conceitos básicos para um melhor entendimento sobre grafos fuzzy, conceitos como: conjuntos fuzzy e tipos de função de pertinência. Um conjunto é definido em [18] como uma tentativa de organizar, resumir e generalizar conhecimento sobre objetos. Neste sentido é trabalhado com uma dicotomia sobre um objeto pertencer ou não a um determinado conjunto. Esta dicotomia pode ser da seguinte forma:

$$Nt(x) = \begin{cases} 1, & se \;\; x \in A \\ 0, & se \;\; x \notin A \end{cases}$$

Exemplo: dado um conjunto de bons alunos de uma instituição com notas $Nt = \{x \in \Re | x \geq 8,0\}$. Os alunos com notas iguais ou superiores a *8,0* são seguramente considerados bons alunos. Já, os alunos com notas *7,9* não podem ser considerados como bons alunos.

Neste caso um subconjunto *fuzzy A* em X é definido por uma *função de pertinência* μA que associa cada ponto de X a um número real no intervalo [0,1], com o valor de μA em x representando o *grau de pertinência* de x em A ($Pr(x)$). Então, quanto mais próximo o valor de $Pr(x)$ estiver da unidade, maior o grau de pertinência de x em A.

Na Figura 5 é mostrado um exemplo de uma função característica clássica e na Figura 6 está o exemplo de uma função de pertinência.

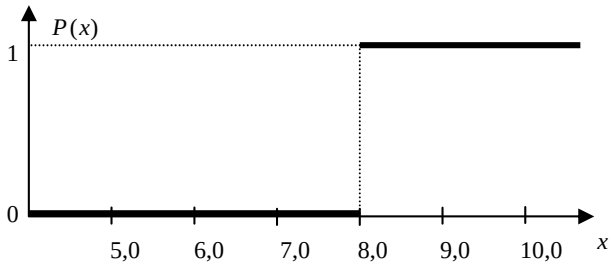

Figura 5: Função característica clássica.

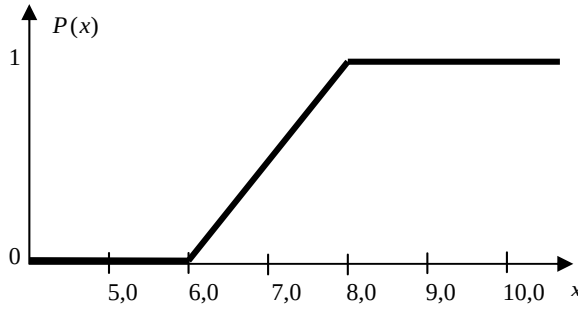

Figura 6: Função de pertinência.

3. Definição do Problema da Árvore Geradora de Custo Mínimo com Grupamentos utilizando Grafos Fuzzy

O problema da Árvore Geradora de custo Mínimo com Grupamentos utilizando Grafos Fuzzy (AGMG-GF), consiste em um problema de grafos, que possui graus de pertinência em suas arestas. Este grau de pertinência também pode ser interpretado como um valor de confiabilidade da solução.

Dado que *G = (V, E)* é um grafo, onde *V* representa um conjunto de vértices, *E* é o conjunto de arestas com um custo $(C \geq 0)$ associado a cada uma delas que ligam os vértices de *V*. O conjunto de vértices *V* é particionado em *k* grupamentos, onde $V = V1 \cup V2 \cup ... \cup Vk$. O objetivo da AGMG é ligar pelo menos um nó (vértice) de cada grupamento (conjunto), de modo que a árvore geradora possa ligar todos os grupamentos do grafo com custo mínimo, onde custo significa a soma dos valores de todas as arestas que formam a AGMG. Isso em um grafo *crisp* com custo *crisp* refere-se ao problema clássico, isto é, um problema em que tanto o grafo quanto os parâmetros associados são bem definidos.

Em um grafo fuzzy, as arestas apresentam graus de pertinência $(\mu \geq 0)$ associado a cada uma delas. Onde este grau de pertinência é interpretado como um valor de confiabilidade da solução. O objetivo da AGMG utilizando grafos fuzzy é ligar pelo menos um nó (vértice) de cada grupamento (conjunto), de modo que a árvore geradora possa ligar todos os grupamentos do grafo com um menor custo e um maior grau de pertinência possível, assim será gerado um conjunto de *soluções fuzzy* $S_0, S_1, ..., Si$, onde o grau de pertinência de cada uma dessas soluções será a menor pertinência encontrada na solução *Si* e, em Si_{+1} não haverá nenhum pertinência menor ou igual às de *Si*. É notável que com o aumento no grau de pertinência deve obter uma árvore geradora de custo mínimo com grupamentos igual ou maior que as anteriormente obtidas.

Com o intuito de esclarecer melhor o problema da Árvore Geradora de custo Mínimo com Grupamentos utilizando Grafos Fuzzy (AGMG-GF) é mostrado um exemplo na Figura 7.

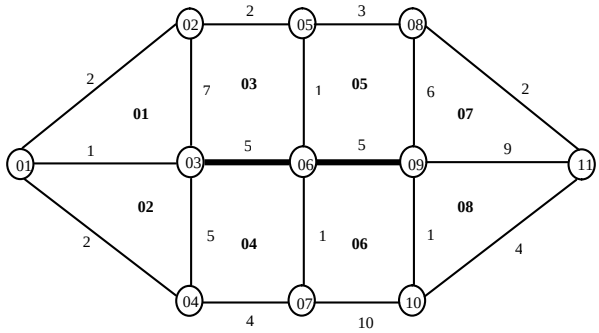

Figura 7: Grafo conexo particionado em oito grupamentos.

Dado um grafo *G=(V,A)* com *m* nós e *n* arestas não direcionado e completo, dividido em 8 grupamentos, o problema consiste em construir uma AGMG, $\overline{T}$, em que a soma dos custos das arestas de $\overline{T}$ seja mínima.

Na Figura 7 é apresentado a Árvore Geradora de custo Mínimo com Grupamentos em linhas destacadas mais grossas, observe-se que foi obtida uma AGMG $\overline{T} = (3,6),\ (6,9)$ com o menor custo, $C = 10$, tocando todos o grupamentos do grafo.

Dado que o grafo da Figura 7 tenha os seguintes graus de pertinências associados em suas arestas, conforme mostrado na Tabela 1.

Tabela 1: Grau de pertinência das arestas do grafo da Figura 7.

Aresta	μ	**Aresta**	μ	**Aresta**	μ
(1,2)	*0,55*	(3,6)	*0,35*	(7,10)	*0,85*
(1,3)	*0,10*	(4,7)	*0,86*	(8,9)	*0,80*
(1,4)	*0,80*	(5,6)	*0,70*	(8,11)	*0,70*
(2,3)	*0,70*	(5,8)	*0,65*	(9,11)	*0,25*
(2,5)	*0,65*	(6,9)	*0,40*	(9,10)	*0,50*
(3,4)	*0,90*	(6,7)	*0,87*	(10,11)	*0,90*

A AGMG obtida do grafo da Figura 7, $\overline{T}_1 = (3,6),\ (6,9)$ é a que possui o menor custo, $C_1 = 10$, mas também possui o menor grau de pertinência $\rho_1 = 0{,}35$. Com a eliminação das arestas $(1,3),(3,6),(9,11)$ que possuem grau de pertinência menor ou igual a ρ_1 é obtida uma nova árvore $\overline{T}_2 = (1,2),\ (2,5),(5,6),(5,8),\ (8,11)$, com o custo, $C_2 = 10$ igual ao custo obtido em $\overline{T}_1$, porém com um maior grau de pertinência $\rho_2 = 0{,}55$. Se retirarmos ás arestas $(1,2),(6,9),(9,10)$ com grau de pertinência menor ou igual a ρ_2 será obtida a árvore $\overline{T}_3 = (2,3),\ (2,5),(5,6),(5,8),\ (8,11)$, que se diferencia de $\overline{T}_2$ pela troca da aresta $(1,2)$ pela $(2,3)$, por ter um maior valor de pertinência $\rho_3 = 0{,}65$ e um maior custo $C_3 = 15$. Eliminando as arestas $(2,5),(5,8)$ com pertinência menor ou igual a ρ_3 obteve-se a arvore $\overline{T}_4 = (1,4),\ (6,7),(4,7),(7,10),\ (10,11)$, com um custo $C_4 = 21$, e pertinência $\rho_4 = 0{,}80$. Retirando as arestas $(1,4),(2,3),(5,6),(8,9),(8,11)$ é obtida uma nova árvore $\overline{T}_5 = (3,4),\ (6,7),(4,7),(7,10),\ (10,11)$, com o custo $C_5 = 24$, e pertinência $\rho_5 = 0{,}85$, onde a aresta $(1,4)$ é trocada pela $(3,4)$. A árvore $\overline{T}_5 = (3,4),\ (6,7),(4,7),(7,10),\ (10,11)$ com o custo $C_5 = 24$ e grau de pertinência $\rho_5 = 0{,}85$ é a árvore com o maior de grau de pertinência obtido do grafo da Figura 7. A partir de agora, a retirada das arestas com graus de pertinências menor ou igual *0,85* tornaria o grafo desconexo e não formaria mais árvores que fossem capazes de tocar todos os grupamentos do grafo.

Do grafo da Figura 7 foram obtidas 5 árvores, ou seja, um conjunto de *soluções fuzzy* que estão apresentadas na Tabela 2, juntamente com seus custos e graus de pertinência.

Tabela 2. Conjunto de soluções fuzzy.

AGMG – GF	**Custo**	μ
(3,6), (6,9)	10	0,35
(1,2), (2,5), (5,6), (5,8), (8,11)	10	0,55
(2,3), (2,5), (5,6), (5,8), (8,11)	15	0,65
(1,4), (4,7), (6,7), (7,10), (10,11)	21	0,80
(3,4), (4,7), (6,7), (7,10), (10,11)	24	0,85

4. GRASP (*Greed Ramdomized Adaptive Search Procedure*)

O GRASP (*Greedy Randomized Adaptive Search Procedure*) que significa em português, "Procedimento de Busca Gulosa Adaptativa Aleatória", é um processo iterativo [12, 13, 14], onde cada iteração GRASP consiste em duas fases, uma fase de construção e uma fase de busca local. Na Figura 8, é apresentado o pseudocódigo do algoritmo GRASP.

```
Procedimento GRASP
1  f(s) ← ∞;
2   Para k de 1 até MaxIter faça
3      x ← GulosoAleatorizado();
4      x ← BuscaLocal(x);
5        Se f (s') < f (s) então
6           s ← s';
7        Fim Se
8   Fim Para
9  Retornar s ;
Fim Procedimento
```

Figura 8: Algoritmo GRASP.

4.1. Fase de Construção

Na fase de construção, a solução viável é construída iterativamente elemento por elemento. A heurística é adaptativa porque os benefícios associados com cada elemento são atualizados a cada iteração da fase de construção para refletir as mudanças ocorridas pela seleção de elementos anteriores. A parte aleatória corresponde à forma de escolha dos melhores candidatos da lista. Cada iteração é composta por três subclasses [5, 16]:

- Construção da Lista Restrita de Candidatos (LRC), a qual contêm um conjunto reduzido de elementos candidatos a pertencer à solução;
- Escolha aleatória do elemento na LRC e inclusão de elemento na solução;
- Adaptação ou recálculo da função gulosa para os elementos ainda não pertencentes à solução.

A melhor solução encontrada ao longo de todas as iterações GRASP realizadas é retornada como resultado. Na fase de construção do GRASP, uma solução viável é construída iterativamente, um elemento da solução por vez, até que a solução esteja completa. Os elementos candidatos que compõem a solução são ordenados em uma lista, chamada de lista de candidatos, a qual contém

todos os candidatos. Esta lista é ordenada por uma função gulosa que mede o benefício que o elemento escolhido mais recentemente concede à parte da solução já construída. Um subconjunto denominado lista restrita de candidatos (LRC) é formado pelos melhores elementos que compõem a lista de candidatos.

O tamanho da LRC é controlado por um parâmetro α, onde para α = 0 tem-se um comportamento puramente guloso do algoritmo e para α = 1, um comportamento aleatório. A componente probabilística do método é devida à escolha aleatória de um elemento da LRC. Este procedimento permite que diferentes soluções de boa qualidade sejam geradas.

Desta forma, o principal parâmetro a ser configurado no GRASP é a cardinalidade da LRC. Este parâmetro é o mais importante para o procedimento GRASP: se |LRC| = 1, então a solução inicial é totalmente gulosa, se |LRC| = *n* então a solução é totalmente aleatória. A média e a variância do valor de função objetivo das soluções construídas são diretamente afetadas por tal parâmetro: se |LRC| é pequena, menor a variância, menor o espaço de soluções percorrido e maior a chance de aprisionar a busca local em um ótimo local pobre; se |LRC| é grande, maior a variância, menor possibilidade de prisão em ótimo local pobre, porém maior número de iterações com soluções sub-ótimas e maiores as vizinhanças exploradas.

Para a aplicação eficaz do método é necessária, portanto a definição de um intervalo de valores para |LRC| de forma a balancear a relação entre qualidade das soluções, quantidade de iterações necessárias e vizinhança explorada [12, 14].

4.2. Fase de Busca Local

Métodos de busca local em problemas de otimização constituem uma família de técnicas baseadas na noção de vizinhança, ou seja, são métodos que percorrem o espaço de pesquisa passando, iterativamente, de uma solução para outra que seja sua vizinha.

A fase de busca local de GRASP aproveita a solução inicial da fase de construção e explora a vizinhança ao redor desta solução. Se um melhoramento é encontrado, a solução corrente é atualizada e novamente a vizinhança ao redor da nova solução é pesquisada. O processo se repete até nenhum melhoramento ser encontrado.
De acordo com [15], é preciso ter cuidado em:

- Escolher uma vizinhança apropriada;
- Usar estruturas de dados eficientes para acelerar a busca local;
- Ter uma boa solução inicial;
- Partir de uma boa solução inicial (perto de um ótimo local), conduz-se a uma busca local eficiente.

5. Algoritmo Proposto

Neste capítulo, é apresentada uma descrição das principais ferramentas e representações do algoritmo (denominado A-MG) proposto neste trabalho para solução do problema da AGMG-GF.

O A-MG foi implementado seqüencialmente na linguagem C, utilizando uma metaheurística de construção e melhoria chamada GRASP, cujo pseudocódigo é descrito na Figura 9.

Procedimento *A-MG_GRASP*
1 $f(S) \leftarrow \infty$;
2 $f(P) \leftarrow \emptyset$;
3 **Enquanto** formar árvore **faça**
4 **Para** *i* de 1 até *MaxIter* **faça**
5 Aplicar a fase de construção para obter uma solução viável *S*,*P* ;
6 Aplicar busca local em *S* gerando uma nova solução S^* , P^* ;
7 **Se** custo de $f(S^*) <= f(S)$ **então**
8 $S \leftarrow S^*$;
9 **Se** μ de $f(P^*) > f(P)$ **então**
10 $P \leftarrow P^*$;
11 **Fim Se**
12 **Fim Se**
13 **Fim Para**
14 Eliminar arestas com $\mu > P$**;**
15 **Fim Enquanto**
16 **Retornar** *S, P*;
Fim Procedimento

Figura 9: A-MG GRASP.

5.1. Algoritmo GRASP

Neste algoritmo, o usuário define o número de execuções que ele deseja. A partir de então, executa-se a fase de construção, que procura construir uma solução viável e de qualidade. Logo em seguida é executada a fase de busca local, procurando refinar a solução inicial. Após a fase de construção e o refinamento da solução (busca local), atualiza-se a melhor solução encontrada, levando em consideração o custo e o grau de pertinência.

A escolha do GRASP deu-se porque o mesmo vem

obtendo bons resultados na resolução de diversos outros problemas combinatórios.

5.2. Fase de Construção

Na fase de construção do algoritmo inicialmente e construída uma Árvore Geradora Mínima (AGM) denominada $\overline{T}$, a partir desta AGM $\overline{T}$ é feita uma poda de todos os nós folhas e que não seja o único de um determinado grupamento, posteriormente a AGM $\overline{T}$ é atualizada com o seu custo e grau de pertinência.

O algoritmo para construção de uma solução inicial viável pode ser descrito conforme a Figura 10.

```
Procedimento A-MG_Construção
1  S ← Ø ; P ← Ø ;
3  Inicializar n_v, n_a, n_g;
4  Ordenar o grafo de forma crescente
   de acordo com valor das arestas;
6  Para i ate n_v -1 faça
7      Criar uma AGM T ;
8  Fim Para
9  Para i ate n_v faça
10     Se Vi é nó folha e não é o único de um grupamento
           então
11         Remover Vi de T ;
12         Atualizar T ;
13     Fim se
14     S ← {custo de T };
15     P ← {grau de pertinência de T };
16 Fim Para
17 n_va ← {numero de vértices de S};
18 Retorne S, P;
Fim Procedimento
```

Figura 10: A-MG Construção.

5.3. Fase de Busca Local

Neste algoritmo *A-MG_Busca_Local* parte-se de uma solução inicial gerada pelo algoritmo *A-MG_Construção*, apresentado na Figura 10. A Figura 11 ilustra o algoritmo *A-MG_Busca_Local*.

A fase de busca local aproveita a solução inicial da fase de construção *A-MG_Construção* e explora a vizinhança ao redor desta solução. Se um melhoramento é encontrado, a solução corrente é atualizada e novamente a vizinhança ao redor da nova solução é pesquisada. O processo se repete até nenhum melhoramento ser encontrado.

```
Procedimento A-MG_Busca_Local
1  S ← {melhor custo da solução inicial} ;
2  P ← {melhor pertinência da solução inicial} ;
3  Para i ate n_v faça
4   Se Vi é nó folha é o único de K grupamento; então
5      Remover de S a aresta A- que possui
       extremidade de V igual a 1;
6      Guardar a qual Ki a mesma pertence;
7   Fim se
8   Para cada Vi restante em S faça
9      Verificar se existir uma aresta A-
       que ligue Vi de S a outro Vj em Ki;
10     Se existir e fornecer um S menor e um
          maior grau de pertinência então
11        Atualizar S ← S* ;
12        Atualizar P ← P* ;
13        Diminuir os graus dos Vi que saem;
14        Aumentar os graus dos Vi que entram;
15     Fim se
16  Fim Para
17 Fim Para
18 Para i ate n_va faça
19   Verificar o menor grau de pertinência de S;
20   P ← {menor grau pertinência de S};
21 Fim Para
22 Retorne S, P;
Fim Procedimento
```

Figura 11: A-MG Busca Local.

Partindo desta solução inicial, verificar se existe algum nó folha que seja o único de algum grupamento. Se existe, retirar da solução *S* a aresta que possui extremidade de vértice igual a 1 e guardar a qual grupamento a mesma pertence. Após isto, verificar se existe uma outra aresta que ligue um vértice de *S* a outro vértice no grupamento se existir e fornecer uma solução menor e um grau de pertinência maior atualizar a solução (custo *S* e *P* grau de pertinência). Por fim, verificar o menor grau de pertinência da árvore, atualizar o grau de pertinência *P*.

6. Resultados Computacionais

Nesta Seção, serão apresentados os testes e resultados computacionais realizados sobre o algoritmo proposto neste trabalho, utilizando a metaheurística GRASP. A seguir, serão apresentadas as instâncias utilizadas para teste, o desempenho do A-MG, tanto sobre a qualidade do conjunto de soluções fuzzy fornecida quanto pelo tempo de execução.

6.1. Testes

O A-MG foi submetido às instâncias disponibilizadas por [3] para o problema da Árvore Geradora de custo Mínimo com Grupamentos (AGMG)

sendo que cada uma das 15 instâncias (gmst1, gmst2,..., gmst15) foram adaptadas para o problema da Árvore Geradora de custo Mínimo com Grupamentos utilizando Grafos Fuzzy (AGMG-GF). Foi adicionada em todas as instâncias, graus de pertinência em suas arestas, assim podendo ser aplicadas ao problema da AGMG-GF.

Cada uma das instâncias é um grafo conexo e particionado em vários grupamentos, onde o número de grupamentos varia de 4 a 50, o número de nós varia de 25 a 500, o número de arestas varia de 50 a 5000 e por fim o grau de pertinência tem valores sorteados em um intervalo entre 0 (zero) e 1 (um). Cada grupamento possui inúmeros vértices conectados entre si, e a cada aresta é atribuído um peso (custo, valor ou distância) e um grau de pertinência. No problema da Árvore Geradora de custo Mínimo com Grupamentos utilizando Grafos Fuzzy o grau de pertinência é interpretado como confiabilidade da solução. Na Tabela 3 está a ilustração das instâncias e todos os seus parâmetros, onde *V* é o numero de vértices, *A* o numero de arestas, *G* números de grupamentos e *P* o grau de pertinência.

Tabela 3: Instâncias Fuzzy.

Instâncias	V	A	G	P
gmst 1	25	50	4	[0 – 1]
gmst2	25	100	8	[0 – 1]
gmst3	25	150	10	[0 – 1]
gmst4	50	150	5	[0 – 1]
gmst5	50	300	10	[0 – 1]
gmst6	75	200	8	[0 – 1]
gmst7	75	300	10	[0 – 1]
gmst8	75	400	15	[0 – 1]
gmst9	100	300	7	[0 – 1]
gmst10	100	500	10	[0 - 1]
gmst11	150	300	8	[0 - 1]
gmst12	150	500	12	[0 - 1]
gmst13	200	500	10	[0 - 1]
gmst14	200	1000	20	[0 - 1]
gmst15	250	500	10	[0 - 1]

Ressalta-se que o número de iterações utilizado no GRASP foi de 1000 para todos as instâncias e que o valor de α foi sorteado aleatoriamente entre 0 e 1 a cada nova iteração do método. O valor de 1000 iterações foi determinado após testes empíricos e foi o que apresentou a melhor relação custo benefício.

6.2. Solução e Tempo de Execução

Todas as instâncias foram executadas num computador com a seguinte configuração: Intel Pentium IV 1.8 Ghz, 256 MB de RAM e utilizando o sistema operacional Conectiva Linux 8+.

Na Tabela 4, é mostrado o conjunto de soluções fuzzy (custo da AGMG-GF e grau de pertinência) encontrada pelo A-MG e respectivamente o seu tempo de execução (em segundos de CPU). Onde ***C*** é o custo, ***P*** o grau de pertinência e ***T*** o tempo de execução.

Tabela 4: Desempenho de A-MG.

Instâncias	C	P	T
gmst1	23	0.39	0.000
	23	0.69	0.180
	38	0.80	0.190
gmst2	**41**	**0.52**	**0.740**
	42	**0.54**	**0.100**
	47	**0.70**	**0.420**
gmst3	36	0.34	1.410
	49	0.56	1.470
	64	0.83	1.470
gmst4	**18**	**0.50**	**0.100**
	18	**0.62**	**0.100**
	31	**0.70**	**0.110**
gmst5	27	0.44	75.450
	40	0.58	3.980
	49	0.72	0.810
gmst6	**55**	**0.31**	**0.280**
	55	**0.61**	**0.290**
	85	**0.63**	**32.820**
gmst7	67	0.32	8.320
	77	0.52	8.320
	100	0.57	8.320
gmst8	**53**	**0.35**	**46.260**
	56	**0.59**	**46.260**
	99	**0.62**	**46.260**
gmst9	37	0.50	0.700
	37	0.54	0.700
	67	0.73	0.220
gmst10	**48**	**0.12**	**9.050**
	50	**0.58**	**2.190**
	56	**0.61**	**4.870**
gmst11	50	0.20	0.310
	50	0.37	6.100
	50	0.48	11.000
gmst12	**75**	**0.18**	**38.990**
	99	**0.35**	**38.990**
	126	**0.42**	**3.080**
gmst13	46	0.24	70.730
	54	0.31	2.910
	60	0.33	2.760
gmst14	**58**	**0.31**	**95.74**
	63	**0.45**	**95.74**
	71	**0.71**	**95.74**
gmst15	60	0.28	3.090
	67	0.38	5.600
	92	0.43	3.140

Aqui, não foi feita nenhuma comparação com outro trabalho, pois até o momento não foi encontrado na literatura trabalho similar a este.

7. CONCLUSÕES

Neste trabalho foi estudado o problema da Árvore Geradora de custo Mínimo com Grupamentos utilizando Grafos Fuzzy (AGMG-GF), como contribuição deste trabalho, desenvolveu-se um algoritmo (A-MG) utilizando a metaheurística GRASP para o problema da AGMG-GF. Embora fácil de descrever, o problema da AGMG-GF é de difícil resolução. Como visto anteriormente, este problema está classificado como um problema NP-Dificil o que limita o uso de técnicas exatas para encontrar a solução para instâncias grandes.

O algoritmo proposto denominado A-MG foi testado em 15 instâncias de [3], cabendo aqui ressaltar que as instâncias foram adaptadas para o problema da AGMG-GF. O A-MG obteve um bom desempenho em termos de qualidades de soluções geradas, principalmente em instâncias de elevadas dimensões. Um outro aspecto relevante foram os tempos computacionais satisfatórios.

O objetivo deste trabalho foi encontrar boas soluções num tempo computacional razoável. O objetivo foi plenamente alcançado com a aplicação da metaheurística GRASP, portanto, conclui se que os resultados obtidos pelo A-MG são animadores, boas soluções numa quantidade de tempo computacional viável.

As recomendações e sugestões para trabalhos futuros sugeridas para este trabalho são no sentido de tornar o algoritmo mais robusto e eficiente, podendo assim obter uma melhoria na solução e no tempo de execução. As mesmas são:

- Implementação de GRASP reativo e utilização de memória, com intuito de se obter soluções ainda melhores.
- Desenvolver uma versão paralela da metaheurística GRASP com o intuito de se obter um melhor desempenho que pode acarretar uma melhora da qualidade das soluções obtidas e a redução do tempo de processamento.

Referências Bibliográficas

[1] BLUM, C. and ROLI, A. *Metaheuristics in Combinatorial Optimisation: Overview and Conceptual Comparison*. Technical Report TR/IRIDIA/2001-13, IRIDIA, Université Libre de Bruxelles, Belgium, 2001.

[2] DORIGO, M. and STUTZLE, T. *The Ant Colony Optimisation Metaheuristics: Algorithms, Applications, and Advances*. Technical Report TR/IRIDIA/2000-32, IRIDIA, Université Libre de Bruxelles, Belgium, 2000.

[3] DROR, M., HAOUARI, M. and CHAOUACHI, J. *Generalized Spanning Trees*. European Journal of Operational Research, 120, p. 583-592, 2000.

[4] FEO, T. A. and RESENDE, M. G. C. *Greedy Randomized Adaptive Search Procedures*. Journal of Global Optimization 6, p. 109-133, 1995

[5] MORELLI C. D. S. e VIEIRA L. T (1999) *Análise de Métodos Heurísticos de Característica Gulosa*. IV Semana Acadêmica do PPGC (Programa de Pós-Graduação em Computação), 1999.

[6] NORONHA, T. F., da SILVA, M. M. e ALOISE, D. J. *Uma Abordagem sobre Estratégias Metaheurísticas*. Revista Eletrônica de Iniciação Científica (REIC), Ano I, Vol. I, No. I, 2001.

[7] OCHI, L., BRUNO, L. e FERNANDO, C. *Técnicas Para Melhorar o Desempenho de Algoritmos Evolutivos: Uma Aplicação Para o Problema de Árvore Geradora de custo Mínimo Com Grupamentos*. III Congresso Brasileiro de Computação. Itajaí-SC, p. 661-672, 2003.

[8] PEREIRA, F. C. *Programação de Horários em Escolas via GRASP e Busca Tabu*. Tese graduação em Engenharia de Produção. Universidade Federal de Ouro Preto, 2003.

[9] POP, P. C. *The Generalized Minimum Spanning Tree Problem*. Twente University Press, 2002.

[10] POP, P. C. *New Models of the Generalized Minimum Spanning Tree Problem*. Journal of Mathematical Modelling and Algorithms, pp 153-166, 2004.

[11] REEVES, C. R. *Modern Heuristic Techniques for Combinatorial Problems. Blackwell Scientific Publications*, Osney Mead, Oxford OX2 0EL, 1993.

[12] RESENDE, M. G. C. and RIBEIRO C. C. *Greedy randomized adaptive search procedures*. In Handbook of Metaheuristics, F. Glover and G. Kochenberger, eds., Kluwer Academic Publishers, pp. 219-249, 2003.

[13] RESENDE, M. G. C. *Greedy Randomized Adaptive Search Procedures (GRASP)*, AT&T Labs Research Technical Report: 98.41.1, 1998.

[14] RESENDE, M. G. C. and RIBEIRO, C. C. *Greedy Randomized Adaptive Search Procedures*. AT&T Labs Research Technical Report, Setembro, 2001.

[15] ROCHA, M. L., JUNIOR, C. L. e BARROS, C. M. P. *Aplicação de uma Heurística GRASP Paralela ao Problema da P-Mediana*. X ENCITA. X Encontro de Iniciação Científica

do Instituto Tecnológico de Aeronáutica, 2004.

[16] SOUZA, M. J. F. *Programação de horários em escolas: uma aproximação por metaheurísticas*, Tese de Doutorado, Programa de Engenharia de Sistemas e Computação, UFRJ, Rio de Janeiro, 2000.

[17] TAKAHASHI, M. T. e YAKAMAMI, A. *Um estudo sobre o problema da árvore geradora mínima com estrutura do grafo fuzzy*. XXXVI Simpósio Brasileiro de Pesquisa Operacional (SBPO XXXVI). São João del-Rei, MG, 2004.

[18] TAKAHASHI, M. T. *Contribuições ao Estudo de Grafos Fuzzy: Teoria e Algoritmos*. Tese de Doutorado, Faculdade de Engenharia Elétrica e de Computação, UNICAMP, Campinas, SP, 2004.