UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO
UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

DIEGO ROCHA LIMA

MODELOS E METAHEURÍSTICAS MULTI-OBJETIVOS PARA O PROBLEMA DA ÁRVORE GERADORA DE CUSTO E DIÂMETRO MÍNIMOS

MOSSORÓ – RN
DEZ / 2012

DIEGO ROCHA LIMA

MODELOS E METAHEURÍSTICAS MULTI-OBJETIVOS PARA O PROBLEMA DA ÁRVORE GERADORA DE CUSTO E DIÂMETRO MÍNIMOS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação – associação ampla entre a Universidade do Estado do Rio Grande do Norte e a Universidade Federal Rural do Semi-Árido, para a obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Dario José Aloise - UERN

MOSSORÓ – RN
DEZ / 2012

FICHA CATALOGRÁFICA

DIEGO ROCHA LIMA

MODELOS E METAHEURÍSTICAS MULTI-OBJETIVOS PARA O PROBLEMA DA ÁRVORE GERADORA DE CUSTO E DIÂMETRO MÍNIMOS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação para a obtenção do título de Mestre em Ciência da Computação.

APROVADA EM: ___ / ___ / _____.

BANCA EXAMINADORA

_____

Prof. Dr. Dario José Aloise – UERN
Presidente (Orientador)

_____

Prof(a). Dr(a). Andréa Cynthia dos Santos - UTT/França
Membro (co-orientador)

_____

Prof. Dr. Hugo Alexandre Dantas do Nascimento - UFG
Membro externo

_____

Prof. Dr. Carlos Heitor Pereira Liberalino – UERN
Membro interno

_____

Prof. Dr. Francisco Chagas de Lima Júnior – UERN
Membro interno

RESUMO

Nesta dissertação o problema da Árvore Geradora de Custo e Diâmetro Mínimo (bi-AGCDM) é investigado. Este problema é NP-difícil e consiste em encontrar soluções que satisfaçam os dois objetivos, custo e diâmetro. Esse problema está relacionado com outros problemas conhecidos como da Árvore Geradora Mínima (AGM), Árvore Geradora Mínima com Restrição de Diâmetro (AGMRD) e Árvore Geradora de Diâmetro Mínimo (AGDM). Problemas em redes de transporte, comunicação, energia ou computadores podem ser modelados pelo bi-AGCDM. Neste trabalho é proposta uma formulação para o bi-AGCDM usando estratégias diferentes para tratar com os dois objetivos. Além das formulações matemáticas este trabalho também apresenta adaptações e implementações dos algoritmos NSGA-II (*Nondominated Sorting Genetic Algorithm*) e MOEA(*Multiobjective Evolutionary Algorithm*), recentemente publicado na literatura, e faz uma comparação dos resultados obtidos pelos mesmos para o problema em apreço. Os experimentos foram realizados com quatro grupos de instâncias, três provenientes de trabalhos importantes da literatura e um novo referencial de instâncias criado para este trabalho. Seus desempenhos foram avaliados através das métricas clássicas: quantidade de elementos na fronteira de Pareto, espalhamento das soluções e hipervolume. Os resultados apresentados em tabelas e gráficos mostram a eficiência do modelo proposto. Mostram também que o NSGA-II implementado para o bi-AGCDM encontrou excelentes soluções, em alguns casos encontrando até o mesmo resultado ótimo obtido na literatura para o AGMRD. Nas comparações também ficou comprovada a eficiência do NSGA-II em relação ao MOEA. O tema abordado no trabalho é recente e deixa uma série de pontos que podem ser trabalhados visando a melhoria dos modelos e algoritmos propostos.

ABSTRACT

In this dissertation the problem of Cost and Diameter Minimum Spanning Tree (bi-CDMST) is investigated. This problem is NP-hard and consists of finding solutions that satisfy both objectives, cost and diameter. This problem is related to other problems known as the Minimum Spanning Tree (MST), Diameter-Constrained Minimum Spanning Tree (DCMST) and Diameter Minimum Spanning Tree (DMST). Problems in networks of transportation, communication, energy or computers can be modeled by bi-CDMST. In this work we propose a mathematical formulation for the bi-CDMST using different strategies to deal with the two objectives. Besides the mathematical formulations this work also presents an implementation of algorithm NSGA-II (*Nondominated Sorting Genetic Algorithm*) and a comparison of its results with the MOEA (*Multiobjective Evolutionary Algorithm*) recently published in the literature. The experiments were conducted with four groups of instances, three important works from the literature and a new benchmark instances created for this work. Their performances were evaluated by classic metrics: number of elements in the Pareto front, spacing of solutions and hypervolume. The results presented in tables and graphs show the efficiency of the proposed model. It also show that the NSGA-II implemented for the bi-CDMST found optimal solutions, in some cases finding optimal result obtained in the literature for AGMRD. In the comparisons were also proven the effectiveness of NSGA-II in relation to the MOEA. The theme discussed in this work is recent and makes a number of points that can be worked to improve the models and the algorithms proposed.

LISTA DE TABELAS

LISTA DE ALGORITMOS

LISTA DE MODELOS

LISTA DE FIGURAS

LISTA DE SIGLAS

AG – Algorithm Genetic

AGDM – Árvore Geradora de Diâmetro Mínimo

AGM – Árvore Geradora Mínima

AGMRD – Árvore Geradora Mínima com Restrição de Diâmetro

Bi-AGCDM – Bi-objetivo Árvore Geradora de Custo e Diâmetro Mínimo

GRASP – Greedy Randomized Adaptive Search Procedure

ILS – Iterative Local Search

LCR – Restricted List of Candidates

MOEA – Multi-objective Evolutionary Algorithm

NSGA-II – Nondominated Sorting Genetic Algorithm

OTT – One Time Tree

RHG - Randomized Greedy Heuristic

VND – Variable Neighborhood Descent

VNS – Variable Neighborhood Search

SUMÁRIO

1. INTRODUÇÃO

A Teoria dos Grafos é um ramo da matemática e computação que estudas as relações em um Grafo $G = (V,E)$. Nele V é definido como um conjunto de vértices e E como um conjunto de arestas. No conjunto V estão definidos pontos e no conjunto E as ligações entre os pontos de V, ou seja, que pontos são adjacentes [FEOFILOFF *et al*, 2011]. O estudo da Teoria dos Grafos ajuda a conhecer melhor essa estrutura e aplicá-las à diversos problemas encontrados na matemática e combinatória, informática e computação, física, biologia, engenharia, pesquisa operacional, muitos ramos da indústria. Todos os problemas encontrados nessas áreas, quando modelados de forma correta com a utilização de grafos, apresentam soluções excelentes.

Vários problemas que envolvem a área de Otimização Combinatória podem ser modelados e resolvidos com o estudo de Grafos, como o Caixeiro Viajante, Fluxo em Redes, Coloração de Mapas, Caminhos de Custo Mínimo entre outros. Neste trabalho o estudo trata de problemas que envolvem Árvores Geradoras, que por definição é uma sub-árvore T de um grafo $G = (V,E)$, que contem todos os vértices de G. Este problema é relativamente simples e polinomial. A maior complexidade do problema se dá quando certas restrições são impostas, como grau, custo, diâmetro entre outros. Essas restrições é que na maioria das vezes torna esse problema complexo, fazendo-se necessário e utilização de meta-heurísticas para resolvê-los.

Este trabalho irá abordar uma versão bi-objetivo do problema, onde mais de uma restrição será imposta. Neste caso o custo e o diâmetro. Esse tipo de abordagem ainda é bastante recente e por isso poucos trabalhos são encontrados na literatura onde fazem uso da versão bi-objetivo.

O bi-objetivo Árvore Geradora de Custo e Diâmetro Mínimos (bi-AGCDM) é uma extensão da Árvore Geradora com Restrição de Diâmetro (AGMRD) [GOUVEIA *et al*, 2011; LUCENA *et al*, 2010]e da Árvore Geradora de Diâmetro Mínimo (AGDM)[BUI *et al*,2004; HASSIN e TAMIR, 1995]. O bi-AGCDM também é referenciado na literatura como Problema da Árvore Geradora de Custo Mínimo e Diâmetro Mínimo [HO, 1991]. O Diâmetro em uma árvore geradora é o número de arestas no maior caminho entre os pares de nós. A AGMRD consiste em encontrar uma árvore geradora com custo total mínimo onde o

diâmetro não excede um valor inteiro positivo dado. A AGDM busca uma árvore geradora (não necessariamente com custo total mínimo) onde o diâmetro seja o menor possível. A bi-AGCDM consiste em encontrar uma árvore geradora com o custo total mínimo e com o diâmetro mínimo. Nós propomos formulações matemáticas para o bi-AGCDM usando estratégias diferentes para tratar com os dois objetivos e um novo referencial de instâncias. Além das formulações matemáticas este trabalho também apresenta adaptações e implementações dos algoritmos NSGA-II (*Nondominated Sorting Genetic Algorithm*) [DEB *et al*, 2012] e MOEA[SAHA e KUMAR, 2011] publicado recentemente na literatura, e faz uma comparação dos resultados obtidos pelos mesmos.

O solver utilizado neste trabalho, CPLEX 12, não trabalha com modelos que contenham duas fuções objetivo. Assim será utilizada a estratégia de otimização em duas fases, onde os objetivos são otimizados por ordem de prioridade. O diâmetro é considerado o objetivo prioritário e portanto minimizado na primeira fase. A segunda fase terá minimização do custo como objetivo, respeitando a restrição de diâmetro encontrada na fase anterior.

O bi-AGCDM é uma versão bi-objetivo cujos objetivos não são conflituosos. Em nosso conhecimento, este é o primeiro trabalho dedicado a modelar matematicamente este problema. De posse dos modelos preliminares será feita uma bateria de testes para testar a viabilidade dos modelos propostos e os resultados apresentados na dissertação, bem como possíveis heurísticas e meta-heurísticas para se trabalhar com instâncias grandes.

O problema bi-AGCDM modela diversas aplicações, entre elas algumas no projeto de redes de computadores, onde todos os vértices devem comunicar-se entre si a custo mínimo, garantindo certo nível de serviço. O diâmetro pode ser associado ao retardo ou à confiabilidade de transmissão na rede. Assim, quanto menor o diâmetro, menores serão os atrasos e melhor será a qualidade de serviço (QoS). Outras aplicações práticas são encontradas na implantação de linhas de trens e metrôs. A infraestrura forma uma árvore geradora, onde os custos correspondem à instalação da rede. Além disto, minimizar o diâmetro significa que os caminhos entre todos os pares de estações será o menor possível, reduzindo o tempo entre essas estações ou terminais e provendo QoS [THOMAS *et al*, 2008]. A grande maioria das aplicações que necessitam de uma infraestrutura de rede inteligente pode ser modelada como um problema bi-AGCDM, sejam elas redes de comunicação, de transmissão de energia, de transporte ou de computadores, entre outras.

O texto deste trabalho está dividido em 7 capítulos da como segue: o Capítulo 2 é dedicado a uma ampla revisão bibliográfica sobre o tema e problemas correlatos; o Capítulo 3

mostra o modelo matemático desenvolvido para o problema; no Capítulo 4 é mostrado os testes e resultados obtidos com o modelo matemático; no Capítulo 5 são descritas a abordagem evolucionária e as metaheurísticas implementadas para o problema; no Capítulo 6 é mostrado os testes e resultados computacionais obtidos com os algoritmos implementados. Por fim no capítulo 7 temos as conclusões do trabalho.

2. REVISÃO BIBLIOGRÁFICA

2.1 Árvore Geradora Mínima (AGM)

Considera-se um Grafo não direcionado, conectado e associado a cada arco algum valor não negativo (custo, tempo etc). O Objetivo é encontrar uma estrutura de conexão (árvore) em que todos os nós se conectem uns aos outros por um único caminho. Essa estrutura deve possuir o menor peso possível, onde o peso é dado pela soma dos pesos das arestas escolhidas (mínima).

Matematicamente tem-se que: dado um grafo $G = (V,E)$ onde V é o conjunto de vértices e E o conjunto de arestas, uma AGM de G é dada por um subconjunto de E que conecta todos os nós de V a um custo mínimo [HO, 1991]. Um princípio fundamental das Árvores Geradoras diz que em uma rede com n nós, requer somente n-1 arcos para fornecer um caminho entre todos os pares de nós. Como os n-1 arcos vão formar a Árvore Geradora, basta encontrar a árvore com o menor comprimento total. Um exemplo é dado a seguir.

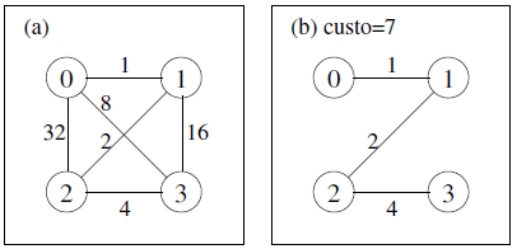


Figura 1 – (a) Grafo(G) com 4 vértices e 6 arestas. (b) - Árvore Geradora Mínima de G

Algumas aplicações de AGM podem ser vistas em:

- Projetos de redes de telecomunicação, redes de computadores, redes de fibra ótica, redes de telefonia, redes de TV a cabo etc;

- Projetos de rodovias e ferrovias;

- Projetos de redes de transmissão de energias

Para este problema existem algoritmos que conseguem resolver o problema da AGM em tempo polinomial. Os mais conhecidos são os algoritmos de PRIM e KRUSKAL, com complexidades $O(n.m)$ e $O(n^2)$, respectivamente.

2.1.1 Algoritmo de KRUSKAL

O algoritmo de KRUSKAL tem como objetivo encontrar uma árvore geradora mínima para qualquer grafo conexo e com pesos associados. Ele parte do princípio de que, se todas as arestas forem sendo escolhidas uma a uma de maneira a não formar ciclos, ao final será formada uma árvore geradora mínima para qualquer grafo. O algoritmo funciona de forma gulosa, ou seja, as arestas escolhidas para entrar na solução são sempre as de menor peso. Sempre que um ciclo é detectado, a aresta é descartada e fica impossibilitada de entrar novamente na solução. O processo se repete até que não haja mais arestas disponíveis. Ao final do algoritmo a árvore estará construída.

Para implementação, este algoritmo utiliza três conjuntos *E, T* e *VS*. *T* é usado para guardar as arestas da árvore expandida. O algoritmo trabalha transformando uma floresta expandida de *G* em uma única árvore. O conjunto *VS* contém os conjuntos das árvores da floresta expandida. Inicialmente *VS* contém todos os vértices de *G*, onde cada vértice é um conjunto unitário.

As arestas são escolhidas por ordem crescente de custo. Quando uma aresta une duas subárvores da floresta expandida, estas são unidas em *VS*, quando não, ela é descartada, pois forma ciclo.

O algoritmo é mostrado com mais detalhes no pseudo-código abaixo.

```
1.  Ler grafo G = (V,E)
2.  Enquanto existirem arestas em E faça
3.      Escolher a aresta de menor peso em E e que não esteja na solução S
4.      Se (não formar ciclo)
5.          Aresta removida de E e inserida em S
6.      Senão
7.          Aresta removida de E
```

```
8.          FimSe
9.      FimEnquanto
10.FimAlgoritmo
```

Algoritmo 1 – KRUSKAL

O algoritmo é bastante simples e de fácil implementação. Uma outra maneira de implementar o algoritmo de KRUSKAL seria ordenar as arestas de forma crescente antes de entrar no laço principal. Assim bastaria percorrer o vetor, ao invés de procurar pela aresta de menor peso. O resultado é exatamente o mesmo e a complexidade também.

Como se pode perceber o algoritmo de KRUSKAL parte de várias árvores desconexas e a solução é montada juntando-se essas sub-árvores até que haja apenas a árvore geradora mínima.

2.1.2 Algoritmo de PRIM

O algoritmo de PRIM é parecido com o anterior, também é guloso, polinomial e de fácil implementação. Este algoritmo agora parte de uma sub-árvore apenas e segue conectando os nós a essa sub-árvore até a árvore geradora mínima seja concluída, diferentemente do anterior que parte de várias árvores desconexas. Ao invés de inserir na solução a aresta de menor peso como em KRUSKAL, no algoritmo de PRIM o próximo nó a entrar na solução é aquele que possui ligação com qualquer nó que já pertença à solução e que tenha o menor peso. Essa estratégia, além de bastante eficiente, é uma maneira de evitar a formação de ciclos.

Para implementação é necessário basicamente a criação de duas estruturas, uma contendo todos os nós do grafo e outra contendo as arestas que estarão na solução final. A cada iteração um nó sai da lista de nós disponíveis e uma de suas arestas entra na solução. Para não formar ciclo, cada nó que ainda não está na solução só pode ser conectado a algum nó que já está na solução. O critério para se escolher a aresta é o menor custo. O processo se repete até que todos os nós tenham sido conectados à AGM que forma a solução final.

O algoritmo é mostrado com mais detalhes no pseudo-código abaixo.

1. Ler grafo $G = (V,E)$
2. Escolhe-se um vértice qualquer (ou aresta de menor peso) e insere o(s) vértices em um conjunto S
3. Retira o vértice de V
4. Enquanto existirem vértices em V faça
5. Dentre os vértices em V seleciona aquele com aresta de menor peso e adjacente a qualquer vértice em S
6. O vértice sai de V e a aresta entra em S
7. FimEnquannto
8. FimAlgoritmo

Algoritmo 2 - PRIM

Assim como o algoritmo de KRUSKAL, o algoritmo de PRIM é simples e de fácil implementação.

2.2 Árvore Geradora Mínima com Restrição de Diâmetro (AGMRD)

O problema da Árvore Geradora Mínima com Restrição de diâmetro (AGMRD) é definido sobre um grafo conexo $G = (V,E)$, onde V é o conjunto de vértices e E é o conjunto de arestas. Cada aresta $[i,j] \in E$ possui custo $c_{ij} \geq 0$ associado. Qualquer árvore geradora de G possui um caminho único P_{ij} que conecta cada par de nós distintos $i,j \in V$. Sendo d_{ij} a quantidade de arestas em P_{ij}, o diâmetro de uma árvore geradora é definido como $\max\{d_{ij}: i,j \in V\}$. O problema da AGMRD consiste em encontrar uma árvore geradora de custo mínimo cujo diâmetro seja menor ou igual a D, onde D é um inteiro positivo satisfazendo $2 \leq D \leq |V| - 1$.

Segundo [SANTOS, 2004] AGMD são polinomiais em quatro casos particulares. O primeiro ocorre quando $D = |V| - 1$ e sua solução corresponde à AGM de G. O segundo caso ocorre quando $D = 2$. Neste caso, a solução é uma árvore em forma de estrela como ilustrado na Figura 2.1-(a). O terceiro caso ocorre quando $D = 3$, cuja solução corresponde a duas estrelas conectadas por uma ponte, chamada neste trabalho de aresta central, ver Figura 2.1-(b). O quarto caso ocorre quando todos os custos das arestas são iguais, não influenciando na construção da árvore. Neste caso, em grafos completos, qualquer estrela é ótima.

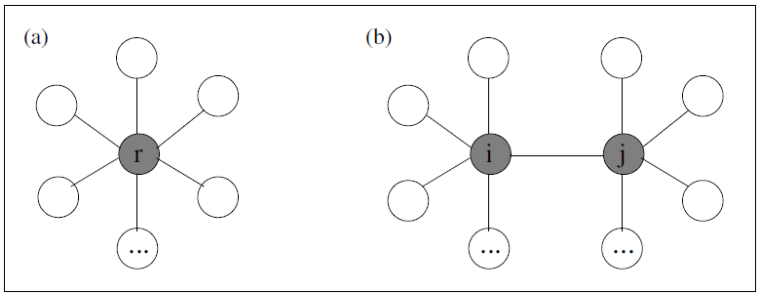


Figura 2 - Casos polinomiais: (a) caso D=2 e (b) caso D=3

2.2.1 Formulações para AGMRD

Formulações matemáticas, heurísticas, meta-heurísticas e algoritmos híbridos têm sido propostas na literatura para o problema da AGMRD. Os modelos em sua maioria fazem a separação dos diâmetros pares e ímpares. Nestes modelos é fixada uma raiz e considera-se que a quantidade máxima de arestas nos caminhos a partir da raiz é igual a $L = D/2$ quando D é par e $L = (D - 1)/2$ quando D é ímpar. Na seqüência serão mostrados dois modelos para a AGMRD, que foram introduzidos inicialmente em [SANTOS *et al*, 2004] e são utilizadas neste trabalho para a segunda fase de otimização.

Estas formulações têm uma propriedade que quando D é par, a AGM tem um vértice central i (respectivamente uma aresta central $e = (i,j)$ quando D é ímpar) em que nenhum outro vértice tem mais que $D/2$ arestas no caminho a partir de i (respectivamente para D ímpar, não mais que $(D-1)/2$ arestas a partir de um das extremidade de e). Um grafo direcionado $G' = (V',E')$ é obtido a partir de $G=(V,E)$ como se segue: um vértice artificial r é

introduzido em *V*. Assim, $V' = V \cup \{r\}$ e $E' = A \cup \{(r,1),...,|V|\}$, e para todos os vértices $[i,j] \in E$, com $i < j$, arcos (i,j) e $(j,i) \in E'$ são adicionados com custos $c_{ij} = c_{ji}$.

Ambas as formulações para *D* par e ímpar usam as variáveis de decisão x_{ij} na escolha do arco $(i,j) \in A'$, e as variáveis não negativas u_i que especificam o número de arcos no caminho de *r* até $i \in V'$. Quando *D* é ímpar, a formulação também usa variáveis binárias z_{ij} que define sempre que um arco $(i,j) \in E$ é selecionado como a aresta central da árvore geradora $z_{ij} = 1$, ou não $z_{ij} = 0$.

Neste trabalho, *D** é dado como sendo o diâmetro ótimo encontrado na primeira fase de otimização. Então $L = D^*/2$ quando *D** é par e $L = (D^* - 1)/2$ quando *D** é ímpar. Abaixo o modelo utilizado para *D** sendo par:

$$\min \sum_{(i,j)\in E} c_{ij} \cdot x_{ij} \qquad st \tag{1}$$

$$\sum_{j\in V} x_{rj} = 1 \tag{2}$$

$$\sum_{(i,j)\in E'} x_{ij} = 1 \quad \forall j \in V \tag{3}$$

$$u_i - u_j + (L+1)x_{ij} + (L-1)x_{ij} \leq L \quad \forall (i,j) \in E' \tag{4}$$

$$u_i \leq L + 1 \quad \forall i \in V' \tag{5}$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in E' \tag{6}$$

$$u_i \geq 0 \quad \forall i \in V' \tag{7}$$

Modelo1 – Formulação para *D par**

O objetivo (1) minimiza o custo total. A restrição (2) garante que o vértice artificial *r* é conectado a apenas um vértice de *V*. A restrição (3) garante que apenas um arco deve ser incidente a cada vértice de *V*. As desigualdades (4) e (5) estabelecem que os caminhos do vértice artificial *r* para cada vértice $i \in V$ tem no máximo $L + 1$ arcos. As variáveis definidas em (6) e (7). Quando *D** é ímpar a formulação tem em conta que a AGM tem uma aresta central com segue:

$$\min \sum_{(i,j)\in A} c_{ij} \cdot x_{ij} + \sum_{(i,j)\in E} c_{ij} \cdot z_{ij} \qquad st \tag{8}$$

$$\sum_{j\in V} x_{rj} = 2 \tag{9}$$

$$\sum_{(i,j)\in E} z_{ij} = 1 \tag{10}$$

$$z_{ij} \geq x_{ri} + x_{rj} - 1 \qquad \forall (i,j) \in E \tag{11}$$

$$z_{ij} \leq x_{ri} \qquad \forall (i,j) \in E \tag{12}$$

$$z_{ij} \leq x_{rj} \qquad \forall (i,j) \in E \tag{13}$$

$$z_{ij} \in \{0,1\} \qquad \forall (i,j) \in E \tag{14}$$

Demais restrições (3) à (7)

Modelo2 – Formulação para *D ímpar**

A função objetivo (8) calcula o custo total incluindo o custo da aresta central. A restrição (9) determina que o vértice artificial *r* é conectado a exatamente dois vértices de *V*. Restrições (10) a (13) dão a linearização de $z_{ij} = x_{ri} \cdot z_{rj}\ \forall (i,j) \in E$. Variáveis z_{ij} são definidas em (25). Restrições (3) a (7) já foram definidas. Para mais detalhes ler a referência [SANTOS *et al*, 2004].

2.3 Heurísticas para AGMRD

As heurísticas para AGMRD tem por objetivo construir uma solução de maneira incremental. Partindo de uma solução vazia, a solução do problema é estruturada utilizando técnicas de adição, agregando novos elementos à solução de forma gradativa a cada iteração, sendo assim, geralmente adotam métodos gulosos e são aplicadas a problemas cujas soluções viáveis são de fácil obtenção.

Na literatura são encontradas duas heurísticas bastante utilizadas para AGMRD. São elas, OTT (*One Time Tree*) e RGH (*Randomized Greed Heuristic*).

2.3.1 Heurística OTT

A heurística OTT procura construir uma solução viável a partir do algoritmo de PRIM, apresentado em 2.2.1. O algoritmo de PRIM obtém uma árvore geradora de custo mínimo através de uma estratégia gulosa. A cada passo é acrescida à arvore a aresta de custo mínimo que não provoca a formação de um ciclo. Além dessa verificação, para que a aresta possa entrar na solução é feito um teste para verificar se o diâmetro não será violado. Em suma, o que acontece em OTT é que a cada passo do algoritmo de PRIM, a aresta só é inserida na solução se, além de não formar ciclo, não violar a restrição de diâmetro.

```
1. Ler grafo G = (V,E)
2. Escolhe-se um vértice qualquer (ou aresta de menor peso) e insere em S
3.      Retira o(s) vértice(s) de V
4.      Enquanto existirem vértices em V faça
5.          Dentre os vértices em V seleciona aquele com aresta de menor peso e adjacente
            a qualquer vértice em S
6.          Se (não violar o diâmetro)
7.              O vértice sai de V e a aresta entra em S
            Senão
8.              Aresta apenas removida de E
9.      FimEnquannto
10.FimAlgoritmo
```

Algoritmo3 – Heurística OTT

Como pode-se perceber, esta heurística possui uma estratégia gulosa já que é baseada no algoritmo de PRIM. Dessa maneira a OTT não tem a garantia de encontrar a solução ótima, mas sim uma solução viável para o problema em questão. Sua complexidade no pior caso é $O(|V|^3)$.

2.3.2 Heurística RGH

Na heurística RGH (*Randomzed Greedy Heuristic*), escolhe-se um nó aleatoriamente a cada iteração. Este nó entra na solução através de uma aresta que o conecta a outro nó que já pertence à solução. Esta aresta deve ser de custo mínimo e não provocar a violação do diâmetro. Ela é aleatória ao ponto que escolhe randomicamente o próximo nó para entrar na solução. Na seqüência ela passar a ser gulosa ao ponto que escolhe a aresta de menor custo que conecta o nó selecionado a algum nó já presente na solução.

```
1. Ler grafo G = (V,E)
2. Escolhe-se um vértice em V (aleatório) e insere em S
3.      Retira o vértice de V
4.      Enquanto existirem vértices em V faça
5.          Dentre os vértices em V seleciona um qualquer (aleatório)
6.          Dentre as arestas em E seleciona a de menor custo que conecte o vértice
            encontrado em 5 à algum presente em S
7.          Se (não encontrar aresta)
8.              Volta ao passo 5
            Senão
9.              Se (não violar o diâmetro)
10.                 O vértice sai de V e a aresta entra em S
                Senão
11.                 Aresta apenas removida de E
12.             FimSe
13.         FimSe
14.     FimEnquannto
15.FimAlgoritmo
```

Algoritmo4 – Heurística RGH

Assim como em OTT a RGH não tem a garantia de encontrar a solução ótima, mas sim uma solução viável para o problema em questão. Sua complexidade é $O(|V|^2)$.

2.4 Metaheurísticas para AGMRD

As meta-heirísticas são amplamente utilizadas para obter soluções em problemas de otimização combinatória e que e que são NP-difíceis. Elas servem para gerar boas soluções, não necessariamente ótimas, em um tempo computacional bem inferior ao requerido se fosse feita uma busca exaustiva passando por todas as possíveis soluções. GRASP (*Greedy*

Randomized Adaptive Search Procedure), PSO (*Particle Swarm Optimization*) e AG (*Algorithm Genetic*) são metaheurísticas bem referenciadas na literatura para o problema da AGMRD.

Nos trabalhos [ARAUJO *et al*,2011; ALNARENGA e ROCHA, 2006; RESENDE e RIBEIRO, 2010] é possível ver implementações, resultados e aplicações do GRASP para o problema em questão. Novas técnicas são inseridas nas fases de busca local do GRASP, como a ILS (*Iterative Local Search*), visando melhorar as soluções. São apresentados métodos alternativos de construção e aceleração da pesquisa. Em comparações vistas nos trabalhos, com VND (*Variable Neighborhood Descent*), VNS (*Variable Neighborhood Search*) e ILS, o GRASP detém os melhores resultados. A utilização da LCR (*Restricted List of Candidates*) também é apresentada neste trabalho.

Em [ERNST, 2010] é apresentada uma heurística nova e híbrida que combina PSO (*Particle Swarm Optimization*) com uma LH (*Lagrangian Heuristic*). Segundo o autor os multiplicadores de Lagrange são utilizados para calcular os limites inferiores e o PSO utilizados para complementar o algoritmo. Com base nas formulações de Lagrange as posições da partícula (i) no PSO são definidas em Multiplicadores de Lagrange e as atualizações de velocidades também combinam o PSO com idéias da LH. O método foi testado com instâncias conhecidas na literatura e apresentou bons resultados.

Os GAs (*Genetic Algorithm*) é a meta-heurística mais bem referenciada na literatura para AGMRD [BINH *et al*, 2008; BINH e NGHIA, 2009; NGHIA *et al*, 2007; LIMA *et al*, 2008; SAHA e KUMAR, 2011; KAMUR *et al*, 2009; BINH *et al*, 2009]. Nestes trabalhos são apresentados novos operadores genéticos e estratégias de recombinação e mutação para o problema. Eles apresentam excelentes resultados em comparação com outros métodos. Inclusive em [KAMUR *et al*, 2009] o autor introduz a "Programação Genética Multi-Objetivo", apresentando sua implementação, testes e resultado afim de comprovar sua eficácia. A partir destes trabalhos pode-se perceber que os algoritmos evolucionários se adéquam muito bem ao problema da AGMRD e possivelmente ao problema tratado neste trabalho: Árvore Geradora de Custo e Diâmetro Mínimos.

As extensões da OTT e RGH também são freqüentemente encontradas em alguns trabalhos [LUCENA *et al*, 2010; ARORA e GARG, 2011; RAQUEJO E SANTOS, 2009; JULSTRON, 2009]. O que eles fazem basicamente é acrescentar novas técnicas em algumas fases da heurística, afim melhorar as soluções encontradas ou acelerar o processo de busca. Essas extensões geralmente estão associadas ao problema em questão, seja pelo tipo de grafo

(euclidiano ou não) ou pela quantidade de arestas sendo mais ou menos denso entre outros. O importante é que cada uma das extensões apresentaram suas vantagens e são facilmente introduzidas nas etapas de construção de meta-heurísticas.

A CP (*Constraint Programming*) é uma abordagem recente para AGMRD [NORONHA *et al*, 2008; NORONHA *et al*, 2010]. Na CP a ideia consiste em gerar restrições em cima de restrições para reduzir o espaço de busca. Ela é capaz de lidar com diâmetros pares e ímpares na mesma formulação. Esta abordagem permite fazer formulações concisas com menor quantidade de variáveis, reduzindo o espaço de busca e assim exigindo menos memória. A otimalidade da técnica é comprovada nos dois trabalhos.

Em [JÚNIOR *et al*, 2011] é apresentada uma aplicação prática da Árvore Geradora Mínima com Restrição de Diâmetro para criar um plano de processamento de consultas *skyline*. A intenção de usar AGMRD é restringir o número de saltos requisitados para a consulta, de modo a limitar a latência do sistema e melhorar o tempo de execução global. Outro objetivo é compartilhar melhor a carga entre os servidores e evitar possíveis pontos de falha.

Nos trabalhos [GRUBER e RAIDL, 2008; GOUVEIA *et al*, 2011] são apresentadas melhorias e restrições válidas para os atuais modelos que permitem resolver o problema da AGMRD. A otimalidade dos modelos é comprovada através dos testes e resultados.

Em [SAFARI e RAHAMATI, 2011] foi proposto um algoritmo baseado na aprendizagem de autômatos para encontrar o resultado ótimo no problema da Árvore Geradora Mínima com Restrição de Diâmetro. Um autômato de aprendizagem é uma unidade de tomada de decisão adaptativa, que melhora o seu desempenho aprendendo a escolher a ação ideal a partir de um conjunto finito de ações permitidas por meio de interações repetidas com um ambiente aleatório. Os resultados do algoritmo proposto foram comparados com ILS, EA, VNS e RGH. A vantagem em relação ao tempo de execução é visível e à medida que a taxa de aprendizagem aumenta o algoritmo se torna mais rápido, devido a maior remoção de bordas.

Estratégias baseadas em agrupamento hierárquico para orientar o processo de construção da Árvore Geradora Mínima com Restrição de Diâmetro são encontradas em [GRUBER e RAIDL, 2009(a); GRUBER e RAIDL, 2009(b)]. As heurísticas de construção dos clusters (agrupamento) são geralmente utilizadas para casos que forma grandes grafos euclidianos, onde o algoritmo tentará encontrar um bom *backbone* (espinha dorsal). A partir desta, é montada a árvore de forma a não exceder o diâmetro.

Por fim, em [GRUBER *et al*, 2006; PATVARDHAN e PRAKASH, 2009] são introduzidas novas estratégias de busca local e algumas heurísticas para o problema da AGMRD. Essas melhorias foram introduzidas em VNS, AG e ACO. Em todos os casos ficou comprovada a melhoria das soluções através da inserção das novas buscas locais.

2.5 Árvore Geradora de Diâmetro Mínimo (AGDM)

O problema da Árvore Geradora de Diâmetro Mínimo (AGDM) consiste basicamente em encontrar o menor diâmetro possível para a Árvore Geradora de um grafo. Este problema ainda recebe pouca atenção na literatura e por isso poucos trabalhos relacionados ao assunto são encontrados [HASSIN e TAMIR, 1995; HO, 1991]. Sua formulação diz que, dado um Grafo não direcionado $G = (V,E)$, V é o conjunto de nós e E é o conjunto de arestas. Também é dito que $|V| = n$ e $|E| = m$. Suponha que cada aresta $e \in E$ está associado com um peso positivo (comprimento) d_e. Uma árvore geradora de G é um subgrafo conectado $T = (V,E)$ sem ciclos. O diâmetro de T, $D(T)$, é definido como o maior dos menores caminhos em T entre todos os pares de nós em V. O problema da AGDM consiste em encontrar uma árvore geradora de G cujo diâmetro seja o mínimo possível.

Em [GOUVEIA e MAGNANTI, 2003] é apresentado um algoritmo novo, que resolve o problema de encontrar uma árvore geradora de diâmetro mínimo distributivamente de qualquer grafo arbitrário com pesos positivos associados ás suas arestas. Foi utilizado um novo protocolo de roteamento dos caminhos mais curtos entre todos os pares de nós (*All-Pairs Shortest Paths* - APSP). O algoritmo resultante distribuído é assíncrono, ele funciona para a rede chamada arbitrária, e alcança $O(n)$ complexidade de tempo e complexidade de mensagens $O(nm)$. Os princípios básicos do algorimo são 3:

1. O cálculo de todos os APSPs em G;
2. O cálculo de um centro absoluto de G;
3. A construção de um MDST de G a partir da tabela APSP e transmissão do conhecimento da MDST a cada processo dentro da rede G

3. MODELO MATEMÁTICO

Este trabalho irá abordar uma versão bi-objetivo do problema de Árvores Geradoras, onde mais de uma restrição será imposta. Neste caso o custo e o diâmetro. Esse tipo de abordagem ainda é bastante recente e por isso poucos trabalhos são encontrados na literatura onde fazem uso da versão bi-objetivo. Será proposta uma formulação matemática para o problema da Árvore Geradora de Custo e Diâmetro Mínimos, bem como uma estratégia meta-heurística para trabalhar com grandes instâncias.

Formalmente, o bi-AGCDM é definido com a seguir. Seja $G = \{V,E\}$ um grafo conectado e não direcionado com um conjunto V de vértices e um conjunto E de arestas. Um custo $c_{ij} \geq 0$ está associado a cada aresta $[i,j] \in E$, com $i < j$. Seja T uma árvore geradora de G. Assim, existe um único caminho P_{ij} em T ligando qualquer par de nós $i,j \in V$. Seja d_{ij} o número de vértices em P_{ij}. Então o diâmetro D de T é definido como $D = max\ \{ d_{ij} : i,j \in V \}$. Além disso, uma Árvore Geradora Mínima (AGM) de G é uma árvore geradora T com custo total mínimo. O bi-AGCDM consiste em encontrar uma AGM com diâmetro mínimo.

O bi-AGCDM é NP-hard [GAREI e JOHNSON, 1979] e em nosso conhecimento, este é o primeiro trabalho dedicado a modelar matematicamente este problema.

3.1 Definição do modelo

A formulação (15)-(25) é inspirada no trabalho de Golveia e Magnanti para a AGMRD [LUCENA *et al*, 2010]. Ele faz uso de um grafo não direcioando $G = (V,E)$ e o diâmetro $max\ \{ d_{pq} : p,q \in V \}$ como já foi mencionado. Sendo c_{ij} as variáveis de decisão na escolha da aresta (i,j). As variáveis de fluxo direcionadas y_{ij}^{pq} especificam se o caminho de $p \in V$ até $q \in V$, com $i \neq q$ e $j \neq p$, passam pela aresta (i,j). As variáveis d_{pq} especificam o número de arestas no caminho de p até q.

$$z_1(x) = \min \sum_{[i,j] \in E} c_{ij} \cdot x_{ij} \tag{15}$$

$$z_2 = \min D \qquad st \tag{16}$$

$$\sum_{[i,j] \in E} x_{ij} = |V| - 1 \tag{17}$$

$$\sum_{j \in V} y_{ij}^{pq} - \sum_{j \in V} y_{ji}^{pq} = \begin{cases} 1, & if\ i = p \\ 0, & i \neq q\ and\ i \neq p \\ -1, & if\ i = q \end{cases} \quad \forall\, i \in V, \forall\, p, q \in V \tag{18}$$

$$y_{ij}^{pq} + y_{ji}^{pq} \leq x_{ij} \qquad \forall [i,j] \in E;\ \forall p, q \in V \tag{19}$$

$$\sum_{[i,j] \in E} \left(y_{ij}^{pq} + y_{ji}^{pq}\right) \leq d_{pq} \qquad \forall p, q \in V \tag{20}$$

$$D \geq d_{pq} \qquad \forall p, q \in V \tag{21}$$

$$y_{ij}^{pq} \in \{0,1\} \qquad \forall [i,j] \in E;\ \forall p, q \in V, i \neq q, j \neq p \tag{22}$$

$$x_{ij} \in \{0,1\}\ \forall [i,j] \in E \tag{23}$$

$$d_{pq} \geq 1 \qquad \forall p, q \in V \tag{24}$$

$$D > 1 \tag{25}$$

Modelo 3 – Formulação geral para bi-AGCDM

Os dois objetivos são dados nas equações (15) e (16) e eles visam respectivamente minimizar o custo total e o diâmetro. A restrição (17) garante que a árvore geradora tem no máximo $|V| - 1$ arestas. As restrições (18) são as clássicas conservações das restrições de multifluxo. As inequações (19) garantem que nenhum fluxo passa através da aresta (i,j) sempre que a aresta (i,j) não pertence à solução, ou seja, $x_{ij} = 0$. Restrições (20) calculam o número de arestas no caminho de p até q. Restrições (21) juntamente com o objetivo (16) minimizam o diâmetro. As variáveis são definidas de (22) à (25).

3.2 Otimização em duas fases

Na utilização de otimização em duas fases, os objetivos são otimizados em ordem de prioridade. Neste caso o diâmetro é considerado o objetivo principal e o custo objetivo secundário. Para um mesmo diâmetro pode-se encontrar várias soluções com custos diferentes. Já o contrário é mais difícil, pois para um mesmo custo não se encontra várias soluções com diâmetros diferentes. Por esso motivo o diâmetro foi minixado pra primeira fase. Esta ação também foi necessária porque os SOLVERS conhecidos até o momento não são capazes de trabalhar com modelos que contenham duas funções objetivo. Nestre trabalhos utilizamos o software CPLEX.

O modelo utilizado para resolver o problema bi-AGCDM foi o apresentado em 3.1. Depois que a primeira fase de otimização é executada temos um diâmetro fixado. A segunda fase de otimização procura uma árvore geradora mínima onde o diâmetro é restrito ao valor encontrado na primeira fase de otimização. Esta estratégia é interessante para aplicações onde menores diâmetros são necessários. Além disso pode-se usar para a segunda fase de otimização formulações já bem conhecidas na literatura como a utilizada neste trabalho e apresentada em 2.3. Ao final das duas fases de otimização o que temos é uma solução (árvore geradora) com menor diâmetro dentre todas as possíveis árvores geradoras e o menor custo possível para aquele diâmetro.

4. TESTES E RESULTADOS

Os experimentos foram realizados em um Intel Core i3 com clock 2.1GHz e 4GB de memória RAM. Foi utilizado o solver CPLEX em sua versão 12.3 e com parâmetos padrão. Os resultados preliminares foram obtidos da otimização em duas fases apresentadas na seção anterior. Por ser um problema novo ainda não há instâncias específicas para o bi-AGCDM. As instâncias encontradas na literatura são para os problemas clássicos da Árvore Geradora Mínima ou Árvore Geradora Mínima com Restrição de Diâmetro. Testes preliminaras foram feitos com tais instâncias, mas não se adequaram bem ao problema. Algumas por serem grandes demais para o modelo e outra por não conseguir fugir dos diâmetros dois e três, que são casos polinomiais. Por isso as mesmas só foram utilizadas nos algoritmos desenvolvidos e mostrados no próximo capítulo. Para este trabalho um novo grupo de instâncias esparsas foi desenvolvido e mostrado a seguir

Um ciclo ou caminho Hamiltoniano arbitrário é construido para garantir que o grafo está totalmente conectado e as arestas restantes são adicionadas aleatoriamente em conformidade com a densidade especificada do grafo. Chamamos de “c” o grupo que contem o cliclo Hamiltoniano e “p” o grupo que contem o caminho Hamiltoniano.

A tabela 1 apresenta os resultados preliminares. O *d* na primeira coluna representa a densidade do grafo. O |*V*| na segunda coluna representa a quantidade de vértices. A quantidade de arestas *E* é dada em função de *d* da seguine forma: $E(d) = \left(\frac{n\,(n-1)}{2} \cdot d\right)$, onde *n* é a quantidade de vértices, ou seja, para um grafo completo tem-se *d* = 1 e para grafos esparsos temos *d* entre 0 e 1. Estes valores são os mesmos para a primeira e segunda fase de otimização. O diâmetro e custo ótimos são representados respectivamente por *D** e *C**. A coluna “RL” representa a relaxação linear, “tempo” representa o tempo que o CPLEX levou para encontrar a solução ótima e “nós” representa a quantidade de nós visitados na árvore *branch and bound* para a correspondente fase de otimização.

				Primeira fase				Segunda fase			
grupo	*d*	|V|	|*E*|	D*	RL	tempo(seg)	nós	C*	RL	tempo(seg)	nós
c	0,2	10	9	9	5	0,36	0	567	500	0.02	0
	0,3	10	13	6	4,04	0,84	21	434	400	0.11	80
	0,4	10	18	4	3,16	2,32	0	305	250	0.09	95
	0,2	15	21	7	4,125	76.9	2375	497	398,3	0.16	171
	0,3	15	31	4	4	74.6	83	635	392,3	1.37	10659
	0,4	15	42	4	3	30862.2	20250	465	341	0.81	2972
	0,2	20	38	5	4	29123.0	14215	980	748,2	0.97	3072
	0,3	20	57	5	3,02	326655.7	72378	618	436,7	1.67	4412
p	0,1	25	30	9	8	41	277	1323	1099	0,16	783
	0,09	25	27	17	11	208	2885	1146	1062	0,13	47
	0,1	30	44	9	7	271062,5	43341	1315	1088	3,9	23736
	0,09	30	39	9	7	49340	32321	1407	1161,1	2,1	10127
	0,08	30	35	12	9	312,5	920	1858	1822,3	0,16	550

Tabela 1 - Resultados para o modelo matemático

Valores ótimos e limites inferiores foram encontrados para as instâncias esparsas. As formulações multifluxo apresentadas em [GOLVEIA e MAGNANTI, 2003] foram adaptadas para o problema da bi-AGCDM. Nos resultados foi possível comprovar a otimalidade do modelo. As densidades foram obtidas de forma experimental, pois alguns valores ficam praticamente inviáveis de serem executados no CPLEX e chegando ao "out of memory", ou seja, estouro de memória.

Os tempos para a primeira fase de otimização são bem mais elevados se comparado com a segunda fase. Este é um motivo que justifica a utilização de meta-heurísticas para os problemas com instâncias que possuem mais que 20 nós e densidade acima de 0.3. Nas segunda fase o modelo já executa com um diâmetro fixo, diminuindo assim o espaço de busca e tornando a execução bem mais rápida que na primeira fase, onde a busca é pelo menor diâmetro possível.

Os resultados obtidos nesta primeira parte do trabalho de dissertação serão de suma importância para a comparação com os resultados obtidos com o Algoritmo Genético desenvolvido na segunda parte do trabalho.

5. ABORDAGEM EVOLUCIONÁRIA

Os algoritmos evolucionários são responsáveis por encontrar as melhores soluções para os problemas que envolvem *multi-objective spannig trees* (MOST)[KUMAR e SINGH, 2007; NEUMANN e WEGENER, 2007]. Essas boas soluções também dependem da codificação utilizada para a representação das árvores geradoras. Neste capítulo mostraremos uma implementação do *fast Nondominated Sorting Genetic Algorithm* (NSGA-II) proposto por [DEB *et al*,2002] que é atualmente um dos métodos mais eficazes para resolução de problemas multiobjetivo, [ZHOU *et al*. 2011]. Para comprovar a eficiência do NSGA-II aplicado ao bi-AGCDM implementamos o M*ulti-objective Evolutionary Algorithm* (MOEA) proposto por [SAHA E KUMAR, 2011] que também é dedicado a resolver problemas bi-objetivo e realizamos as devidas comparações.

5.1 Algoritmo Evolucionário NSGA-II

O NSGA-II é uma metaheurística baseada em algoritmos genéticos destinada a resolver problemas bi-objetivo. A estratégia possui complexidade assintótica de pior caso $O(MN^2)$, onde M´é o número de objetivos e N é o tamanho da população. A idéia é classificar as fronteiras utilizando dois procedimentos: o primeiro ordena as soluções não-dominadas (chamada de *ranking*) e o segundo, estima a densidade de soluções no entorno de um indivíduo (chamado de *crowding distance*).

As fronteiras são camadas no espaço de soluções que limitam uma região de dominância. Na prática diversos objetos $f_m(x), m = 1, \dots, M$, onde $x \in \Re^n$ (vetores que definem soluções do sistema), podem ser considerados, ocupando diversas dimensões no espaço. Sem perda de generalidade, pode-se considerar que todos os objetos de minimização $min[\, f_1\,(x),\ f_2(x), \dots, f_m(x)]$. A Figura 1 exemplifica fronteiras $F_1, F_2, \dots, F_j$ no espaço de soluções para dois objetivos de minimização. Na prática, as fronteiras podem ser descontínuas e formar camadas irregulares. A camada mais importante é a que forma o envelope do espaço de soluções dos diversos objetivos, chamada de fronteira de Pareto. Na primeira fronteira

estão todas as soluções não-dominadas, ou seja, quando não há nenhuma solução melhor que estas. Na segunda fronteira estão as soluções que são dominadas por apenas uma outra solução, e assim sucessivamente. No exemplo da Figura 1 corresponde à fronteira F_1. As soluções Pareto-ótimas são aquelas que não se pode melhorar em nenhum dos dois objetivos e que formam o espaço de soluções não-dominadas.

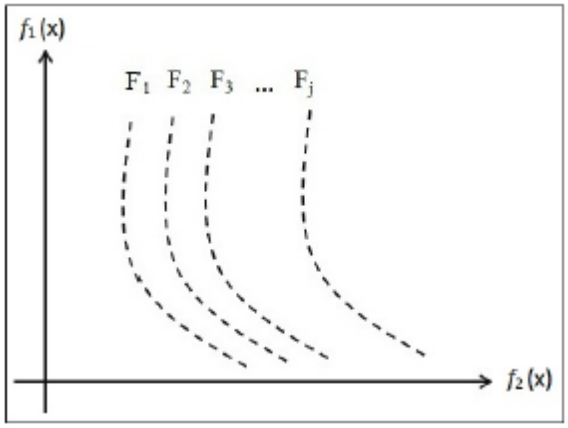


Figura 3 - Camadas ilustrando as fronteiras

Uma solução x_i domina x_j se as seguintes condições são satisfeitas: (a) $f_m(x_i) \leq f_m(x_j)$, para todo $m = 1, \ldots, M$, ou seja, não piora nenhum dos objetivos e (b) $f_m(x_i) < f_m(x_j)$ para algum $m = 1, \ldots, M$, ou seja, é estritamente melhor em pelo menos um objetivo. Isso que dizer que uma solução só domina outra se for melhor em todos os objetivos, ou melhor em apenas um objetivo sendo estritamente igual nos outros.

O procedimento geral do NSGA-II é ilustrado na Figura 2, onde *t* refere-se a iteração corrente. Seja uma população P_t de tamanho *N* formada por indivíduos gerados aleatoriamente e ordenados usando o *ranking*. Os outros *N* indivíduos da população Q_t são obtidos a partir de operadores genéticos, tais como cruzamento e mutação, utilizando indivíduos escolhidos aleatoriamente do conjunto P_t. A população completa R_t é dada por $R_t = P_t \cup Q_t$, onde $|R_t| = 2N$.

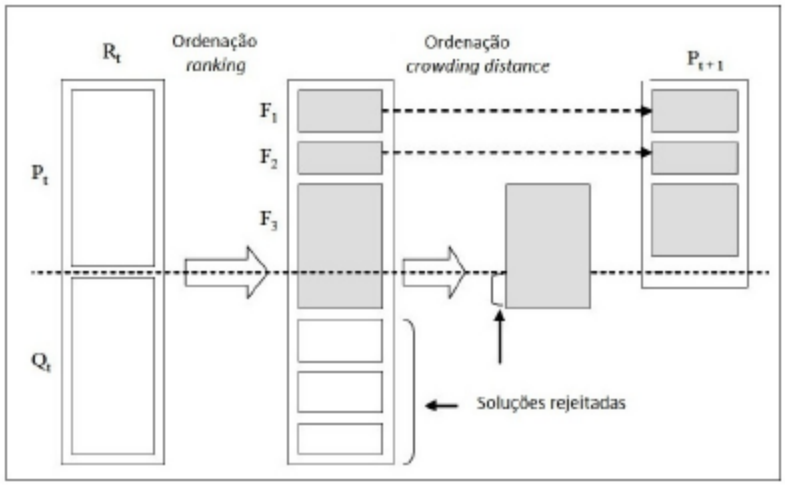


Figura 4 - Procedimento geral do NSGA-II extraido de Deb et al. (2002).

Os procedimentos *ranking* e *crowding distance* são aplicados à população R_t para classificar as fronteiras $\{F_1, F_2, F_3, \ldots, F_j\}$ utilizando o operador de *crowded*, detalhado abaixo. Os primeiros *N* elementos sobrevivem e compõem $P_{(t+1)}$. Operadores genéticos são aplicados, como mencionados acima, para gerar $Q_{(t+1)}$ e o procedimento é repetido até satisfazer um critério de parada.

Dado um indivíduo *j*, a medida *ranking* r_j corresponde a quantidade de soluções que domina *j* na iteração *t*. Logo todos os indivíduos que possuem $r_j = 0$ são armazenados em F_1; os indivíduos com $r_j = 1$ são armazenados em F_2 e assim por diante. No final de uma iteração do NSGA-II todos os indivíduos da população encontram-se classificados em uma fronteira. O objetivo do *crowding distance* é definir a distância dos pontos na vizinhança do indivíduo *j*, na fronteira considerada. Assim sendo, quanto maior for o valor do *crowding distance*, mais distantes estão os pontos, indicando a necessidade de gerar mais soluções nesta região. Seja o vetor de *crowding distance d* de tamanho *M*.

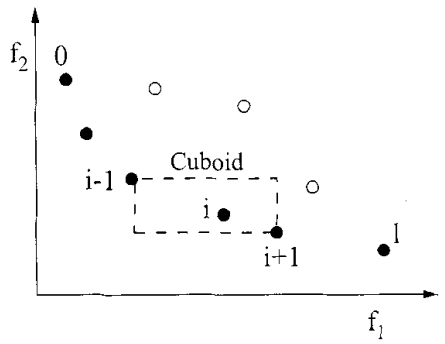


Figura 5 - crowndig distance. Retirado de Deb *et al* (2002)

O cálculo do *crowding distance* é dado na Equação (26), onde *suc(j)* e o *pred(j)* correspondem respectivamente aos valores do sucessor e predecessor do indivíduo *j* no vetor *d*. f_m^{max} e f_m^{min} são os valores máximos e mínimos da função da função objetivo *m*. Detalhes do algoritmo (inicialização do vetor *d*, tratamento e último elemento de *d*, etc) são encontrados em [DEB *et al*, 2002].

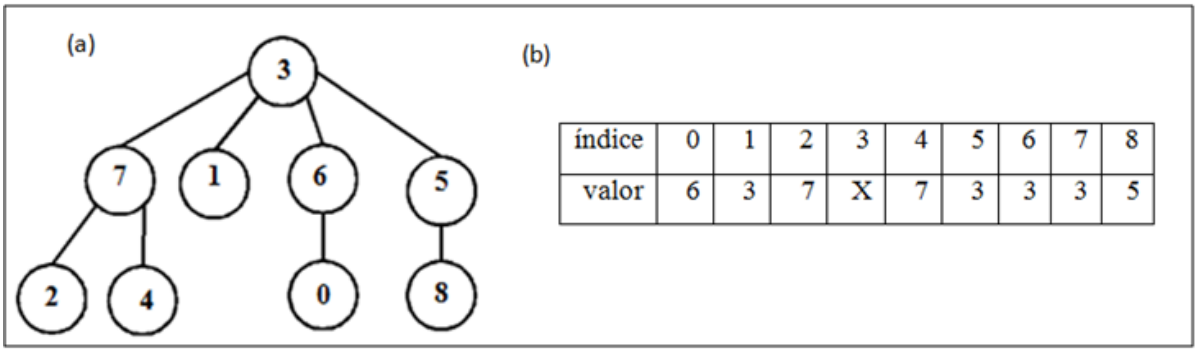
$$d_j = \sum_{m=1}^{M} \left(\frac{f_m^{suc(j)} - f_m^{pred(j)}}{f_m^{max} - f_m^{min}} \right) \tag{26}$$

Nas iterações do NSGA-II o *crowding distance* contribui para fazer a população convergir em direção a F_1, uniformizando a fronteira de Pareto. Após o cálculo do *ranking* e *crowding distance* a população é reordenada utilizando o operador *crowded* definido da seguinte forma: dados dois indivíduos i e j, i domina j se $(r_i < r_j)$ ou $\left((r_i = r_j) \; e \; (d_i > d_j) \right)$. A nova população é construída com os *N* primeiros indivíduos selecionados após o uso do operador *crowded*, formando assim o conjunto $P_{(t+1)}$. Os outros *N* indivíduos, pertencentes ao conjunto $Q_{(t+1)}$, são resultantes da aplicação de operadores genéticos a indivíduos selecionados aleatoriamente do conjunto $P_{(t+1)}$.

5.2 NSGA-II aplicado ao bi-AGCDM

As funções objetivo para o bi-AGCDM minimizam o custo de o diâmetro, ver equações (15) e (16). No NSGA-II desenvolvido para o bi-AGCDM, uma solução (indivíduo) é caracterizada por uma estrutura de árvore geradora, o valor total do custo da árvore e seu diâmetro correspondente. Existem diversas maneiras de representar (codificar) uma árvore em algoritmos genéticos. Alguns trabalhos avaliam a eficiência das diferentes representações em algoritmos genéticos [RAIDL E JULSTRON, 2003a; CARRANO *et al*, 2007]. Neste trabalho um indivíduo é representado através de sua lista de predecessores que refere-se ao nó predecessor na árvore. A Figura 6 mostra um exemplo da codificação utilizada. Em (a) temos

um exemplo de árvore geradora e em (b) sua respectiva codificação com a lista de predecessores. Para se obter os custos basta associar o índice ao seu valor no vetor e buscar na matriz de custos do grafo original. A quantidade de indivíduos gerados é N = |*V*|



índice	0	1	2	3	4	5	6	7	8
valor	6	3	7	X	7	3	3	3	5

Figura 6 - (a) árvore geradora. (b) exemplo de codificação lista de predecessores

Na primeira população duas soluções particulares são geradas e correspondem a melhor solução em termos de custo (AGM) e diâmetro (AG-DM). Para obter a AGM de *G* basta calculá-la utilizando um algoritmo clássico como o de Prim. A AG-DM de *G* é realizada utilizando uma adaptação do algoritmo de busca em largura. Com essa busca é possível determinar qual o menor diâmetro possível em um grafo para todas as suas árvores geradoras. Com o diâmetro de *G* determinado basta escolher um dos caminhos obtidos durante a busca para construir a árvore. As $N - 2$ soluções da população inicial que formam o conjunto P_t são gerados utilizando uma versão aleatória do algoritmo de Prim definida em [RAIDL e JULSTROM, 2003]. Na versão original de Prim a cada iteração o nó selecionado é aquele que se conecta com o menor custo à um outro já pertencente a solução. Na versão aleatorizada um sorteio é feito para selecionar um dos possíveis nós que se conectam a algum outro da solução. A conexão não é necessariamente a de menor custo.

Os indivíduos que fazem parte do conjunto Q_t são gerados utilizando o operador de cruzamento proposto por [CARRANO *et al*, 2007]. O cruzamento consiste em construir uma nova árvore a partir de dois indivíduos selecionados aleatoriamente no conjunto elite P_t. Entretanto, a probabilidade de gerar árvores inviáveis é muito alta, em particular nos grafos esparsos. [CARRANO *et al*, 2007] propôs realizar união de duas árvores que consiste em construir um grafo de suporte contendo todas as arestas das duas árvores. Em seguida, calcular o novo indivíduo a partir do grafo suporte. Por exemplo, dados suas soluções como

mostrado na Figura 7-(a) e 7-(b), a união que forma o grafo de suporte é dado na figura 7-(c). A nova árvore é obtida utilizando o algoritmo aleatorizado de Prim, chamado PrimRST e já mencionado acima.

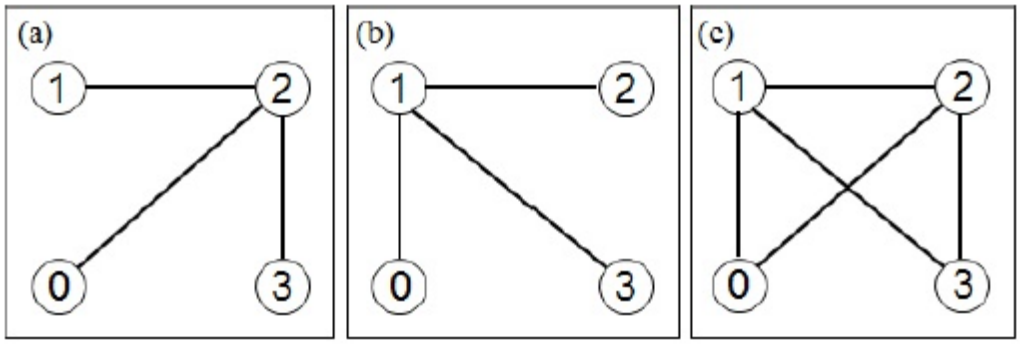


Figura 7 - Exemplo da união de grafos

O *ranking* e o *crowding distance* são calculados para a população inicial e classificados utilizando o operador *crowded*. A construção da população é realizada como mencionado em 6.1.

5.3 Multi-objective Evolutionary Algorithm (MOEA)

O *multiobjective Evolutionary Algorithm* (MOEA) proposto por [SAHA E KUMAR, 2011] é uma metaheurística dedidaca a resolver problemas bi-objetivo em árvores geradoras. Sua codificação e operadores já são específicos para de trabalhar com árvores. O "*edge-set-encoding*" é utilizado para representar suas árvores geradoras. Segundo os mesmos uma das vantagens de se usar MOEAs seria o não fato de não requerer conhecimento específico do problema. A convergência deste MOEA é monitorada pelo "*rank histogram*".

O MOEA proposto por [SAHA E KUMAR, 2011] utiliza um operador de recombinação e dois operadores simples de mutação, chamados de "*greedy edge replacement mutation*" and "*edge-delete mutation*".

O operador de recombinação utilizado é uma adaptação do proposto por [RAIDL E JULSTRON, 2003b]. O operador seleciona duas soluções aleatórias da população elite. Na

sequência cria dois conjuntos de arestas, um F_1com aquelas pertencentes a ambos os pais e o outro F_2 com as arestas pertencentes a apenas um dos pais. A idéia principal é inserir na nova solução primeiramente e sempre que for possível as arestas contidas em F_1. Sempre que não for possível, ou não tiver mais arestas em F_1, inserse-se uma aresta contida em F_2. Durante o processo é feita a verificação para que não haja ciclos e se tenha uma árvore inválida. O processo se repete até que todos os vértices estejam na solução.

Um dos operadores de mutação utilizado é o "*Greedy Edge Replacement Mutation Operator*". Este operador deleta uma aresta randomicamente da solução e insere outra de menor custo de forma a juntar as duas sub-árvores e criar uma nova solução válida. Tem *O(n).*

O segundo operador é o "*Modified Edge Delete Mutation Operator*". Este operador consiste simplesmente em deletar uma aresta ramdomicamente criando assim uma árvore inválida. Logo em seguida é aplicado o operador de recombinação desta árvore geradora inválida com uma outra árvore geradora válida presente na população elite. Após a recombinação teremos novamente uma árvore geradora válida.

De maneira geral, o algoritmo pro posto por Saha e Kumar funciona da seguinte forma: depois de lido o grafo inicial é gerada uma população inicial de tamanho *N*. Feito isso a população é organizada pelo *rank* de dominação e pelo *rank-histogram*. A cada iteração do MOAE dois novos indivíduos são gerados utilizando os operadores de recombinação e mutação apresentados acima.

Os novos indivíduos são inseridos na população e a mesma é reordenada pelo *rank* de dominação. Os dois piores indivíduos são descartados e o *rank-histogram* é novamente calculado. O processo se repete até que o critério de parada seja satisfeito, neste caso, quando a convergência monitorada pelo *rank-histogram* chegar ao ponto desejado. O pseudo-código do MOEA proposto por Saha e Kumar é mostrado a seguir.

1. $t \leftarrow 1$
2. Gerar a população inicial R_t;
3. Calcular os valores dos objetivos para cada indivíduo $i \in R_t$;
4. Classificar cada indivíduo $i \in R_t$ de acordo com seu nível de dominação
5. Ordenar a população pelo rank de dominação
6. Enquanto (critério de Parada não for atendido) faça
7. $t \leftarrow t + 1$
8. Selecionar indivíduos do conjunto R_t
9. Aplicar os operadores genéticos e gerar dois novos indivíduos
10. Calcular os valores dos objetivos para os novos indivíduos $i \in R_t$;
11. Ordenar a população pelo rank de dominação
12. Retirar os dois piores indivíduos da população
13. FimEnquanto
14. Fim

Algoritmo 5 – Procedimento geral do MOEA

O procedimento geral é simples e como pode-se perceber é rápido. A eficiência do MOEA é garantida pela qualidade de seus operadores

5.4 MOEA aplicado ao bi-AGCDM

Neste trabalho foi necessário fazer algumas adaptações ao MOEA, para que se possa ser aplicado ao bi-AGCDM e conseqüentemente uma comparação justa com o NSGA-II proposto nesse trabalho. Uma das modificações consiste no critério de parada. No NSGA-II foi feita uma calibração para se poder definir uma quantidade razoável de iterações, onde a fronteira de Pareto convergisse de forma eficiente e sendo este o critério de parada. No MOEA não seria possível fixar uma quantidade de iterações, já que o critério de para do mesmo está baseado no *rank-histogram*, que é outro método que avalia a convergência da população. Dessa maneira, decidimos adotar o mesmo critério de parada para ambos como

sendo o tempo de execução. Os algoritmos iniciam sua execução e quando atingir um tempo limite preestabelecido o algoritmo encerra sua execução retornando todas as soluções contidas na fronteira de Pareto até o momento, ou seja, a fronteira com todas as soluções não dominadas.

Outra modificação feita ao MOEA se deu em relação a representação (codificação) da árvore. No MOEA original a representação é o "*edge-set encoding*" e neste trabalho implementamos o algoritmo com uma lista de predecessores que refere-se ao nó predecessor na árvore, assim como no NSGA-II.

Como mostrado no algoritmo do MOEA, a cada iteração dois novos indivíduos são gerados e mais dois são descartados. Observamos que isso deixava a convergência do algoritmo muito lenta. Então, a fim de comparação com o NSGA-II, decidimos por gerar N indivíduos a cada iteração e descartar outros N, assim como no NSGA-II.

Os demais procedimentos são exatamente os mesmos do artigo original de [SAHA E KUMAR, 2011]. Os operadores de mutação e recombinação foram utilizados sem modificações.

6. TESTES E RESULTADOS

Os algoritmos desenvolvidos neste trabalho, NSGA-II e MOEA, foram todos desenvolvidos utilizando a linguagem de programação C ANSI com o compilador DevCpp versão 4.9.9.2. Decidimos por essa linguagem devido a sua robustez, e por possibilitar menores tempos de compilação e de execução do que outras. Todos os testes foram realizados em uma máquina Intel core i3 com 2.1GHz e 4GB de memória RAM. Quatro grupos de instâncias são utilizados nos experimentos, onde três *L*, *R*, *G* são respectivamente provenientes dos trabalhos de [LUCENA *et al* ,2010; RAIDL e JULSTRON, 2003a; GOUVEIA e MAGNANTI, 2003]. O novo grupo de instâncias *S* contém grafos esparsos provenientes de [SANTOS *et al*, 2012], cuja conectividade do grafo é respectivamente garantida pela construção de um ciclo ou caminho Hamiltoniano entre os nós do grafo. As outras arestas são selecionadas aleatoriamente e adicionadas de acordo com a densidade do grafo requerida.

O cálculo das métricas de avaliação é uma parte importante do trabalho, pois servem para se avaliar a eficiência das soluções e a qualidade dos algoritmos. Três métricas de avaliação bem conhecidas para problemas bi-objetivos são empregadas nos testes e são calculadas a partir das soluções obtidas na última iteração do algoritmo, NSGA-II ou MOEA. Nas soluções observa-se que os valores de custo são bem mais elevados que os valores de diâmetro. Isso deixa os resultados um tanto estranhos já que as métricas variam bastante de um grupo de instâncias pra outro. Assim fizemos um ajuste e normalizamos todos os valores antes de calcular suas métricas. Todas as soluções tiveram seus valores de custo e diâmetro normalizados entre 0 e 1.

A métrica Q indica a quantidade de indivíduos da fronteira F_1.

A segunda métrica *Spacing*, referida neste trabalho por *S*, foi proposta por [VELDHUIZEN e LAMONT, 2000] e é baseada no espalhamento das soluções. Para obter *S*, é feito um cálculo com base na média das distâncias Euclidianas entre todos os pontos vizinhos na F_1. Sua fórmula é detalhada a seguir:

$$S \triangleq \sqrt{\frac{1}{n-1}\sum_{i+1}^{n}(\bar{d} - d_i)^2}$$

onde: $d_i = min_j\ (|\int_1^i(\vec{x})\ - \int_1^j(\vec{x})\ |\ + |\int_2^i(\vec{x})\ - \int_2^j(\vec{x})\ |)$, $i, j = 1, \ldots, n, \bar{d}$ é a média de todos os d_i, e n é o número de valores na fronteira de Pareto. O valor 0 nesta métrica indica que todos as soluções na fronteira de Pareto estão eqüidistantes.

A métrica *H*, [ZITZLER *et al*, 2003], representa o Hipervolume fornecendo a área coberta pelas soluções presentes na F_1 em relação a um ponto *w* específico. Pode-se considerar o ponto *w* como sendo a pior solução conhecida no espaço de busca. Neste trabalho definimos *w* como sendo uma solução formada pelos piores valores em termos de custo e diâmetro, ou seja, custo da AG-DM e o diâmetro da AGM. Quanto maior o Hipervolume, melhor serão as soluções porque indicam que a fronteira F_1está mais afastada de *w*. Um exemplo do hipervolume de uma solução é mostrado na figura abaixo:

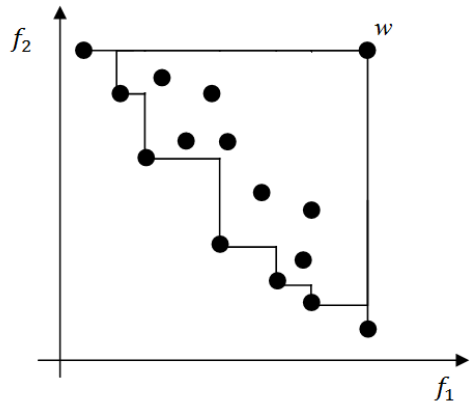


Figura 8 - Hipervolume de uma fronteira de Pareto

6.1 Calibração dos algoritmos

Testes preliminares foram realizados com os 4 grupos de instâncias. Utilizamos apenas duas instâncias de cada grupo para esta calibração dos algoritmos. O objetivo inicial era verificar a convergência das soluções produzidas pelo NSGA-II e pelo MOEA. Três execuções diferentes foram realizadas variando a quantidade de iterações em 100, 300 e 500. A semente fornecida para geração de números aleatórios nas três execuções são idênticas e iguais a 7. Os resultados são mostrados na tabela a seguir. Na primeira coluna temos a

indicação de qual algoritmo foi utilizado. Na segunda coluna temos a especificação dos grupos de instâncias e na terceira qual instância foi utilizada. Q, S e H estão representando especificamente quantidade de indivíduos na fronteira 1, *Spacing* e Hipervolume. O tempo está representado em segundos. Cada um destes resultados estão divididos em 100, 300 e 500 iterações.

			Iteração 100				Iteração 300				Iteração 500			
	Grupo	Instancia	Q	S	H	tempo	Q	S	H	tempo	Q	S	H	tempo
NSGA-II	L	c_v20_a190	9	0,254	0,775	0,146	10	0,239	0,776	0,452	11	0,162	0,787	0,781
		c_v25_a300	12	0,175	0,745	0,242	12	0,175	0,746	0,756	12	0,175	0,746	1,320
	R	c_v70_d7_1	16	0,127	0,800	4,749	18	0,120	0,832	17,799	19	0,126	0,836	31,973
		c_v100_d10_1	14	0,105	0,782	12,797	28	0,078	0,863	52,025	30	0,080	0,873	99,635
	S	Hc_v35_e47	8	0,113	0,666	1,052	7	0,121	0,686	3,467	7	0,121	0,686	5,789
		Hc_v40_e78	9	0,130	0,769	1,411	10	0,134	0,776	4,588	10	0,143	0,781	7,636
	G	se_v40_a400	14	0,115	0,808	1,084	14	0,115	0,817	3,957	14	0,110	0,818	6,735
		se_v60_a600	17	0,104	0,852	3,949	20	0,098	0,859	14,046	21	0,097	0,860	24,192
MOEA	L	c_v20_a190	6	0,201	0,549	0,094	9	0,118	0,610	0,300	8	0,116	0,636	0,464
		c_v25_a300	4	0,454	0,356	0,157	5	0,373	0,461	0,409	5	0,364	0,476	0,614
	R	c_v70_d7_1	8	0,282	0,383	4,264	10	0,247	0,495	12,391	10	0,198	0,567	27,096
		c_v100_d10_1	6	0,483	0,284	14,623	8	0,411	0,361	39,778	8	0,398	0,392	68,012
	S	Hc_v35_e47	6	0,181	0,540	0,675	8	0,093	0,547	2,078	8	0,095	0,584	3,303
		Hc_v40_e78	5	0,379	0,352	1,156	5	0,413	0,374	3,506	4	0,485	0,425	5,367
	G	se_v40_a400	7	0,249	0,556	0,760	10	0,238	0,641	2,248	8	0,286	0,649	3,194
		se_v60_a600	12	0,210	0,609	2,639	11	0,257	0,678	7,344	11	0,262	0,688	1,215

Tabela 2 - Resultados de calibração

A medida que a quantidade de iterações aumenta o hipervolume também cresce. Isso indica que a distância da F_1 em relação ao ponto *w* está aumentando e as soluções estão convergindo e formando cada vez mais uma fronteira de Pareto com melhores soluções. A quantidade de elementos na fronteira F_1 não necessariamente indica que uma fronteira é melhor que a outra, pois podem acontecer casos onde uma solução domina duas ou mais, melhorando a fronteira e diminuído a quantidade de elementos na F_1. O *Spacing* também é outra métrica que não necessariamente diz se uma fronteira é melhor que outra, já que ela indica apenas a uniformidade da fronteira, independentemente da quantidade de elementos. Abaixo é exibido um gráfico que mostra a convergência da F_1 ao decorrer das iterações utilizando o algoritmo NSGA-II.

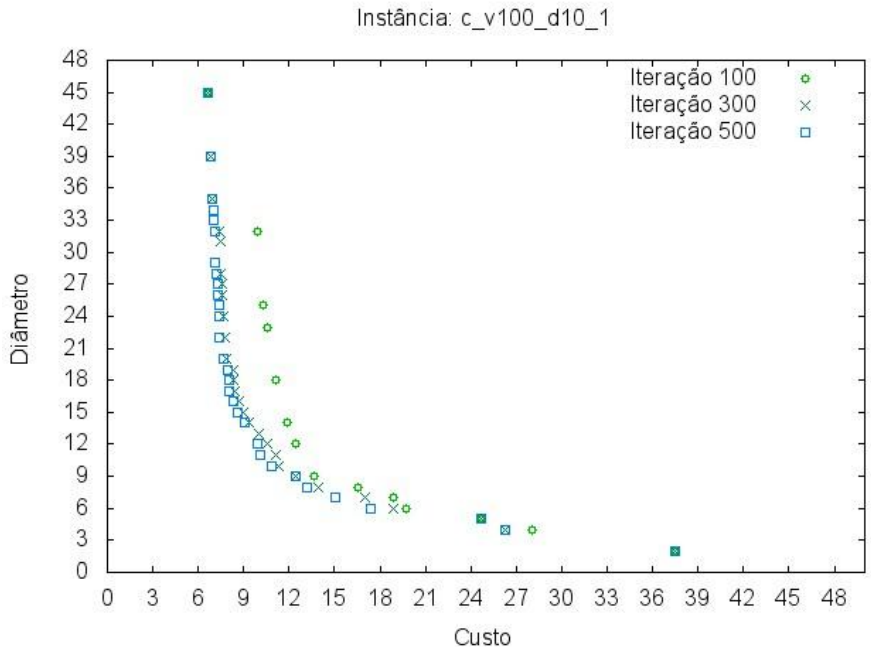


Figura 9 - Evolução da fronteira de Pareto com NSGA-II

Na figura 9 observa-se que a F_1 evoluiu ao decorrer das iterações. Com 100 iterações as soluções são piores do que com 300 iterações e estas por sua vez piores do que com 500 iterações. Pode-se perceber que de 300 para 500 iterações as soluções melhoram pouco, diferentemente do que acontece de 100 para 300 iterações. Assim, acima de 500 iterações as soluções melhoram pouco, entrando na fronteira F_1 cada vez menos soluções a cada iteração.

Decidimos assim que 500 iterações seria uma quantidade suficiente para encontrar boas soluções no NSGA-II para o bi-AGCDM.

6.2 Resultados preliminares

Após a calibração, e decidido que 500 iterações seria uma quantidade suficiente de iterações do NSGA-II para se obter boas soluções, executamos todas as instâncias dos 4 grupos já mencionados anteriormente, usando como critério de parada 500 iterações.

As figuras seguintes mostram o resultado para uma instância de cada grupo. Vale salientar que os resultados foram similares para todas as instâncias do grupo.

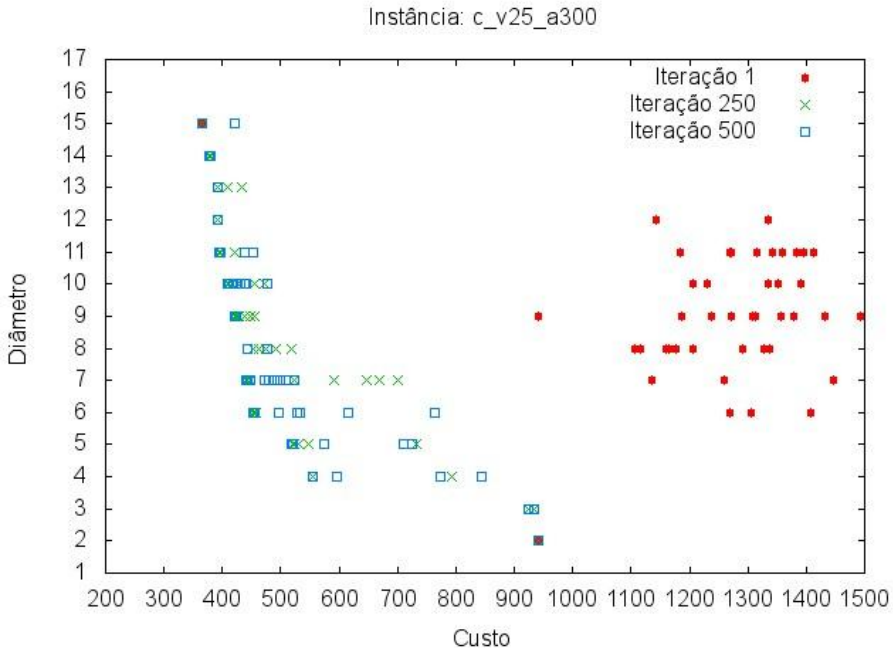


Figura 10 - Evolução da população para instância c_v25_a300

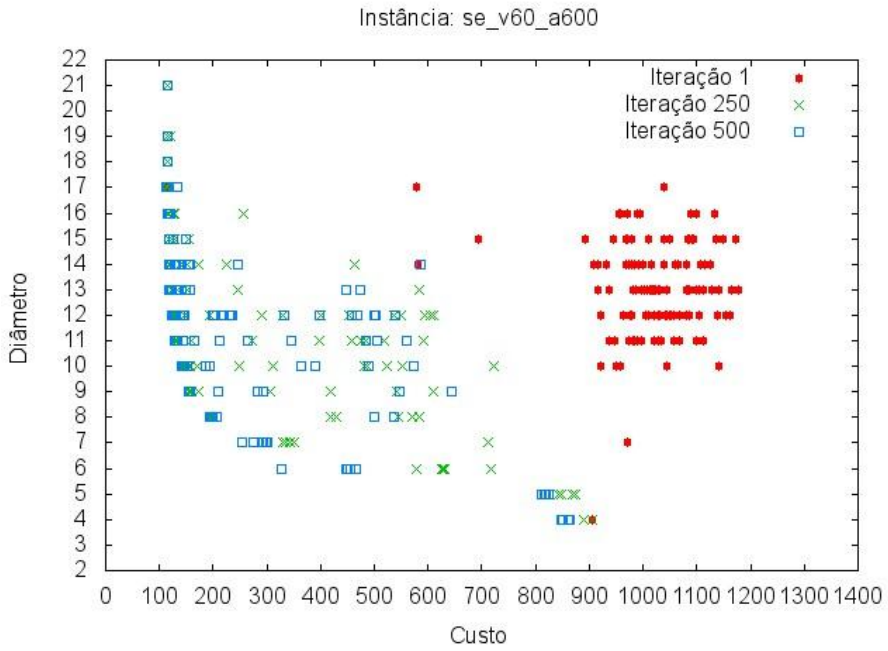


Figura 11 - Evolução da população para instância se_v60_a600

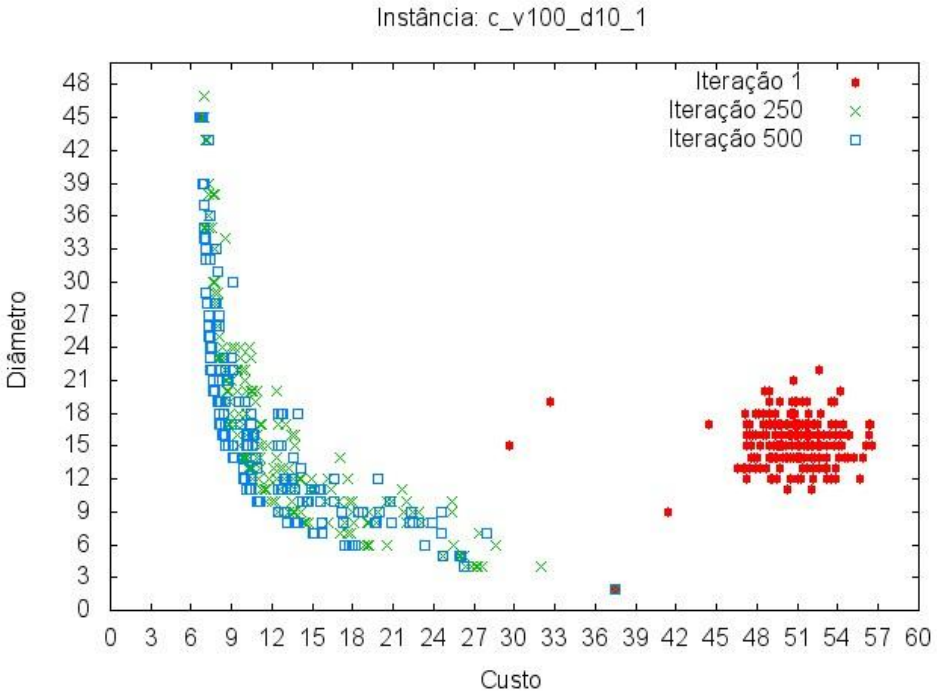


Figura 12 - Evolução da população para instância c_v100_d10_1

As instâncias apresentadas nas figuras anteriores forma escolhidas porque representam um comportamento similar ao que ocorreu no seu respectivo grupo de instâncias. A população gerada na primeira iteração encontra-se concentrada em uma região do espaço de busca de pouca qualidade. Isso ocorre porque *N* árvores são geradas aleatoriamente a as outras *N* são obtidas a partir do cruzamento das árvores aleatórias. A convergência é realizada rapidamente. Na metade das iterações, já existem diversas soluções que se encontram na F_1, e quando não ocorre as soluções já estão bastante próximas.

Em termos de processamento vemos que o algoritmo NSGA-II é eficiente, executando suas 500 iterações em poucos segundos. Nas instâncias mais complexas, como os grafos completos de 100 nós as 500 iterações foram executadas na faixa de 100 segundos.

6.3 Comparação NSGA-II e MOEA

Nos experimentos de comparação algumas adaptações ao MOEA foram necessárias para se fazer uma comparação justa entre os dois algoritmos. As modificações foram detalhadas em 6.1. O MOEA usa como critério de parada o *rank-histogram*, enquanto o NSGA-II desenvolvido neste trabalho utiliza a quantidade de iterações. Assim, considerou-se o mesmo critério de parada para ambos como sendo o tempo de execução. O tempo foi obtido de forma experimental. Deixamos o NSGA-II executando até que encontre algumas das soluções ótimas obtidas na literatura para o problema da "Árvore Geradora Mínima com Diâmetro Restrito", ou que sua fronteira se aproxime das soluções ótimas conhecidas na literatura. Definido esses tempos, demos mais alguns segundos para arredondar os valores e executamos todas as instâncias dos 4 grupos, deixando sempre o mesmo critério de parada no NSGA-II e MOEA para uma mesma instância. Assim como na calibração os valores estão normalizados entre 0 e 1. As métricas de avaliação são exatamente as mesmas dos testes preliminares, quantidade de soluções na F_1, *Spacing* entre as soluções da F_1 e Hipervolume das soluções da F_1. Os resultados para todas as instâncias dos 4 grupos são mostrados a seguir. Cada um das tabelas mostra o seu respectivo grupo de instâncias. Na coluna 1 tem-se a indicação da respectiva instância, seguida por V, representando a quantidade de vértices, A representando a quantidade de arestas, Q, S e H representando respectivamente a quantidade

de nós na F_1, *Spacing* e Hipervolume. A última coluna representa o tempo em segundos definida para a instância.

RAIDL e JULSTRON, 2003a									
			NSGA-II			MOEA			
Instância	V	A	Q	S	H	Q	S	H	tempo
c_v50_d5_1	50	1225	22	0,063	0,744	14	0,160	0,773	60
c_v50_d5_2	50	1225	22	0,131	0,839	18	0,115	0,724	60
c_v50_d5_3	50	1225	18	0,114	0,795	12	0,149	0,645	60
c_v50_d5_4	50	1225	16	0,119	0,696	11	0,141	0,566	60
c_v50_d5_5	50	1225	14	0,084	0,745	12	0,128	0,628	60
c_v70_d7_1	70	2415	18	0,101	0,808	12	0,145	0,629	120
c_v70_d7_2	70	2415	29	0,128	0,891	13	0,204	0,760	120
c_v70_d7_3	70	2415	24	0,092	0,846	15	0,174	0,761	120
c_v70_d7_4	70	2415	23	0,110	0,811	9	0,233	0,596	120
c_v70_d7_5	70	2415	21	0,099	0,814	14	0,107	0,689	120
c_v100_d10_1	100	4950	30	0,096	0,877	8	0,302	0,312	180
c_v100_d10_2	100	4950	29	0,094	0,883	10	0,292	0,640	180
c_v100_d10_3	100	4950	18	0,135	0,841	9	0,307	0,617	180
c_v100_d10_4	100	4950	26	0,104	0,840	8	0,279	0,489	180
c_v100_d10_5	100	4950	25	0,096	0,865	10	0,332	0,711	180
c_v250_d15_1	250	31125	17	0,212	0,762	2	0,000	0,000	450
c_v250_d15_2	250	31125	21	0,163	0,838	2	0,000	0,000	450
c_v250_d15_3	250	31125	14	0,258	0,717	3	0,325	0,224	450
c_v250_d15_4	250	31125	19	0,184	0,781	5	0,503	0,324	450
c_v250_d15_5	250	31125	16	0,174	0,807	6	0,398	0,296	450

Tabela 3 - Resultados para instâncias do RAIDL e JULSTRON, 2003a

SANTOS *et al*, 2012									
			NSGA-II			MOEA			
Instância	V	A	Q	S	H	Q	S	H	tempo
c_v10_a45_d10	10	45	6	0,241	0,662	6	0,143	0,630	10
c_v10_a45_d4	10	45	7	0,122	0,623	7	0,143	0,600	10
c_v10_a45_d5	10	45	4	0,289	0,545	3	0,677	0,232	10
c_v10_a45_d6	10	45	4	0,283	0,548	4	0,508	0,462	10
c_v10_a45_d7	10	45	7	0,140	0,642	7	0,141	0,608	10
c_v10_a45_d8	10	45	5	0,168	0,562	5	0,168	0,562	10
c_v15_a105_d10	15	105	10	0,176	0,782	8	0,252	0,718	15
c_v15_a105_d4	15	105	10	0,176	0,782	8	0,252	0,718	15
c_v15_a105_d5	15	105	10	0,176	0,782	8	0,252	0,718	15
c_v15_a105_d8	15	105	6	0,236	0,650	5	0,272	0,546	15
c_v25_a300_d4	25	300	12	0,169	0,793	12	0,160	0,716	25
c_v25_a300_d5	25	300	12	0,192	0,810	11	0,209	0,703	25
c_v25_a300_d6	25	300	13	0,175	0,808	12	0,175	0,720	25
c_v25_a300_d8	25	300	13	0,174	0,835	11	0,140	0,734	25
c_v25_a300_d9	25	300	9	0,190	0,733	8	0,237	0,615	25
s_v20_a50_d4	25	300	8	0,181	0,747	7	0,201	0,741	20
s_v20_a50_d5	25	300	8	0,242	0,752	7	0,343	0,657	20
s_v20_a50_d6	25	300	7	0,155	0,651	7	0,150	0,599	20
s_v20_a50_d7	25	300	7	0,287	0,770	6	0,158	0,605	20
s_v20_a50_d8	25	300	7	0,200	0,662	6	0,158	0,505	20
s_v40_a100_d4	25	300	17	0,156	0,850	12	0,186	0,836	30
s_v40_a100_d5	25	300	12	0,095	0,794	8	0,328	0,707	30
s_v40_a100_d6	25	300	9	0,148	0,758	8	0,227	0,640	30
s_v60_a150_d5	25	300	13	0,143	0,733	7	0,293	0,496	40

Tabela 4 - Resultados para instâncias de SANTOS *et al*, 2012

LUCENA *et al*, 2012									
			NSGA-II			MOEA			
Instância	V	A	Q	S	H	Q	S	H	tempo
Hc_v10_e13	10	13	2	0	0	2	0	0	5
Hc_v10_e18	10	18	2	0,000	0,000	2	0,000	0,000	5
Hc_v15_e21	15	21	3	0,290	0,638	2	0,000	0,000	10
Hc_v15_e31	15	31	5	0,093	0,599	5	0,075	0,590	10
Hc_v15_e42	15	42	5	0,229	0,588	5	0,113	0,574	10
Hc_v20_e38	20	38	4	0,135	0,405	4	0,248	0,289	15
Hc_v20_e57	20	57	3	0,469	0,484	2	0,000	0,000	15
Hc_v20_e76	20	76	6	0,174	0,611	5	0,111	0,493	15
Hc_v25_e120	25	120	7	0,188	0,691	7	0,305	0,688	20
Hc_v25_e60	25	60	6	0,139	0,593	6	0,344	0,578	20
Hc_v25_e90	25	90	4	0,314	0,696	3	0,523	0,285	20
Hp_v25_e27	25	27	2	0,000	0,000	2	0,000	0,000	20
Hp_v25_e30	25	30	6	0,136	0,659	6	0,153	0,657	20
Hp_v30_e34	30	34	4	0,128	0,588	4	0,293	0,494	25
Hp_v30_e39	30	39	5	0,131	0,445	4	0,060	0,300	25
Hp_v30_e43	30	43	7	0,114	0,661	6	0,229	0,516	25
Hp_v35_e47	35	47	6	0,157	0,767	8	0,206	0,646	30
Hp_v35_e53	35	53	1	0,000	0,000	2	0,000	0,000	30
Hp_v35_e59	35	59	6	0,292	0,709	5	0,234	0,622	30
Hp_v40_e62	40	62	4	0,283	0,582	5	0,211	0,358	35
Hp_v40_e70	40	70	4	0,267	0,551	4	0,509	0,453	35
Hp_v40_e78	40	78	9	0,157	0,766	8	0,197	0,292	35

Tabela 5 - Resultados para instâncias do LUCENA *et al*, 2012

GOUVEIA e MAGNANTI, 2003									
			NSGA-II			MOEA			
Instância	V	A	Q	S	H	Q	S	H	tempo
se_v40_a400_d5	40	400	13	0,126	0,810	10	0,185	0,816	60
se_v60_a600_d5	60	600	19	0,149	0,876	10	0,274	0,757	100
sr_v40_a400_d5	40	400	8	0,156	0,649	7	0,143	0,612	60
sr_v60_a600_d5	60	600	13	0,198	0,788	6	0,175	0,481	100

Tabela 6 - Resultados para instâncias do GOUVEIA e MAGNANTI, 2003

De acordo com os resultados pode-se observar uma melhor eficiência do NSGA-II em relação ao MOEA para o problema em questão. Em todos os casos o hipervolume obtido pelas soluções do NSGA-II é superior ao obtido no MOEA, mostrando assim que a Fronteira de Pareto obtida pelo nosso algoritmo consegue avançar mais e se distanciar mais do ponto *w* que o MOEA. Com relação ao *spacing*, medida que avalia a uniformidade das soluções, o mesmo pode ser observado. Como o NSGA-II na grande maioria dos casos consegue encontrar mais soluções na sua fronteira de Pareto, a mesma fica mais uniforme que no MOEA. Observa-se também que em todos os casos o *spacing* obtido no NSGA-II se aproxima mais de 0 que no outro algoritmo. A quantidade de soluções na fronteira não necessariamente indica qual dos dois algoritmos obteve melhor desempenho. Quando se usa o critério de não dominância para classificar soluções podem ocorrer casos onde ao decorrer das iterações uma única solução domina duas ou mais. Isso pode ser observado por exemplo no resultado das instâncias Hp_v35_e47 e Hp_v40_e62, onde no MOEA a fronteira tem uma ou duas soluções a mais, embora o hipervolume mostre que a fronteira do NSGA-II é superior em termos de soluções eficientes.

A seguir criamos alguns gráficos que mostram as fronteiras obtidas pelo NSGA-II e pelo MOEA. No mesmo gráfico iserimos algumas soluções ótimas obtidas através de modelos exatos e provenientes do trabalho **[**GOUVEIA, SIMONETTI e UCHOA, 2009] para o problema da "Árvore Geradora Mínima com Restrição de Diâmetro" (AGMRD), O intuito é observar o quão próximo das soluções ótimas, as soluções obtidas pelos algoritmos estão.

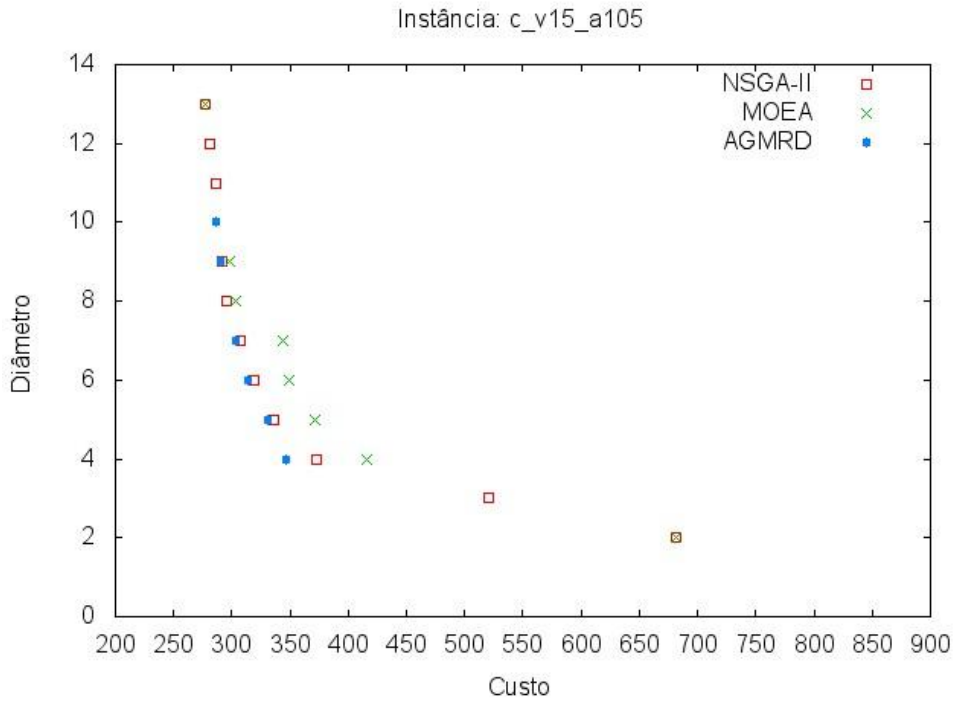


Figura 13 - NSGA-II, MOEA e soluções ótimas para instância c_v15_a105

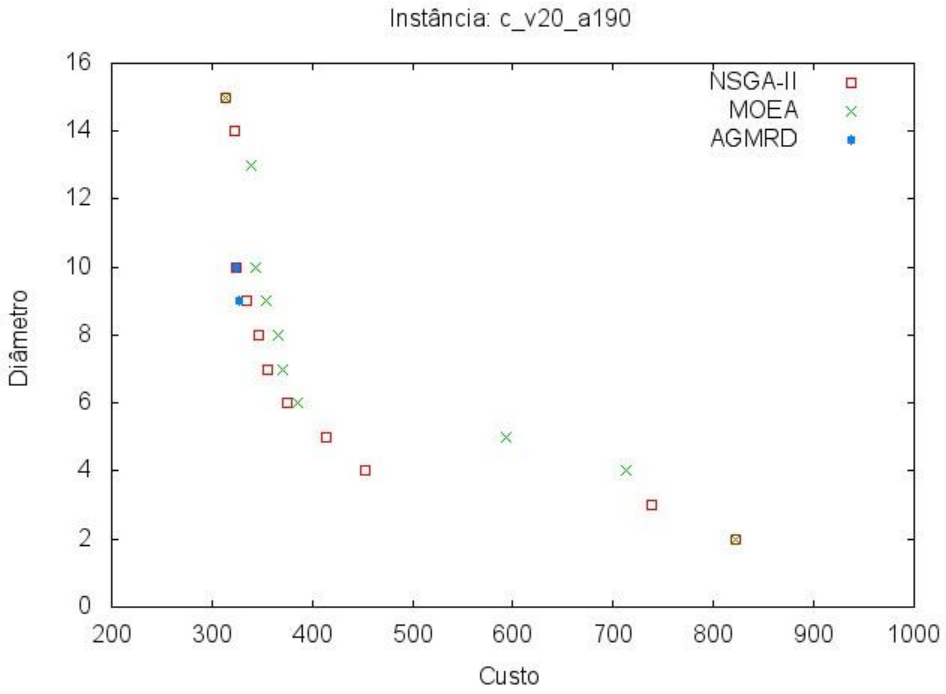


Figura 14 - Figura 13 - NSGA-II, MOEA e soluções ótimas para instância c_v20_a190

Nos gráficos pode-se observar que mesmo não encontrando a solução ótima para todos os diâmetros, a fronteira obtida pelo NSGA-II se aproxima bastante das soluções ótimas, encontrando a mesma solução para o AGMRD em alguns casos. Vale salientar que as soluções obtidas para o AGMRD foram encontradas através de modelos exatos além do que esse problema possui o diâmetro restrito, ou seja, pré-definido, diminuindo assim seu espaço de busca. O NSGA-II diferentemente busca os menores custos para todos os diâmetros possíveis e que pertençam a sua fronteira de Pareto. A medida que o diâmetro aumenta pode-se perceber que as soluções do NSGA-II se aproximam casa vez mais das soluções ótimas. Isso ocorre porque a quantidade de combinações para os diâmetros maiores é bem menor que para os diâmetros menores, facilitando assim a busca. Resultados similares são encontrados para todas as instâncias dos demais grupos. A seguir demais gráficos onde podemos ver a comparação entre NSGA-II e MOEA.

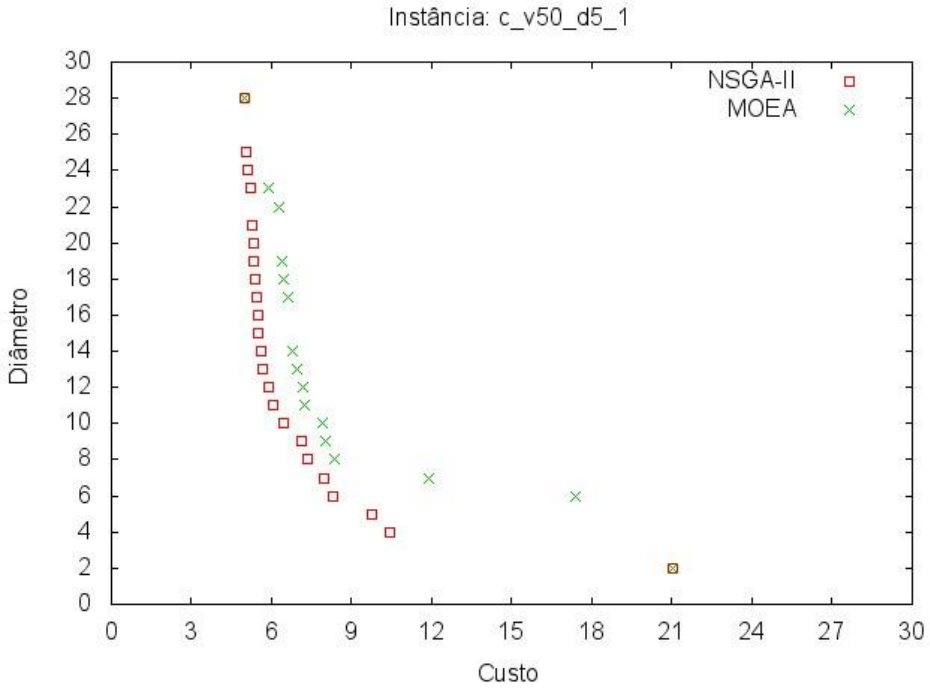


Figura 15 - Fronteira de Pareto encontrada pelo NSGA-II e MOEA para instância c_v50_d5_1

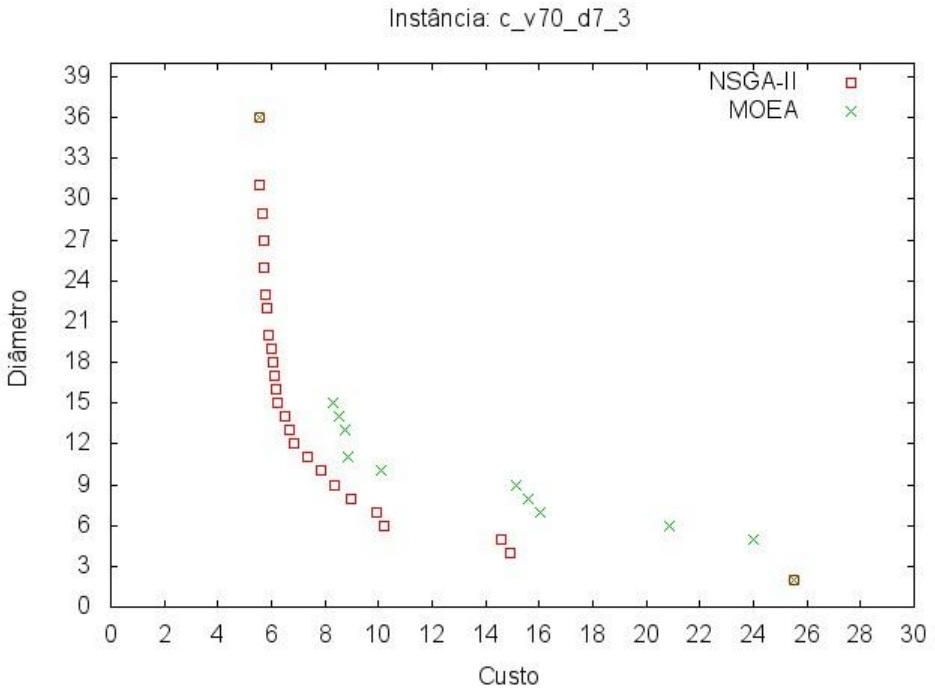


Figura 16 - Fronteira de Pareto encontrada pelo NSGA-II e MOEA para instância c_v70_d7_3

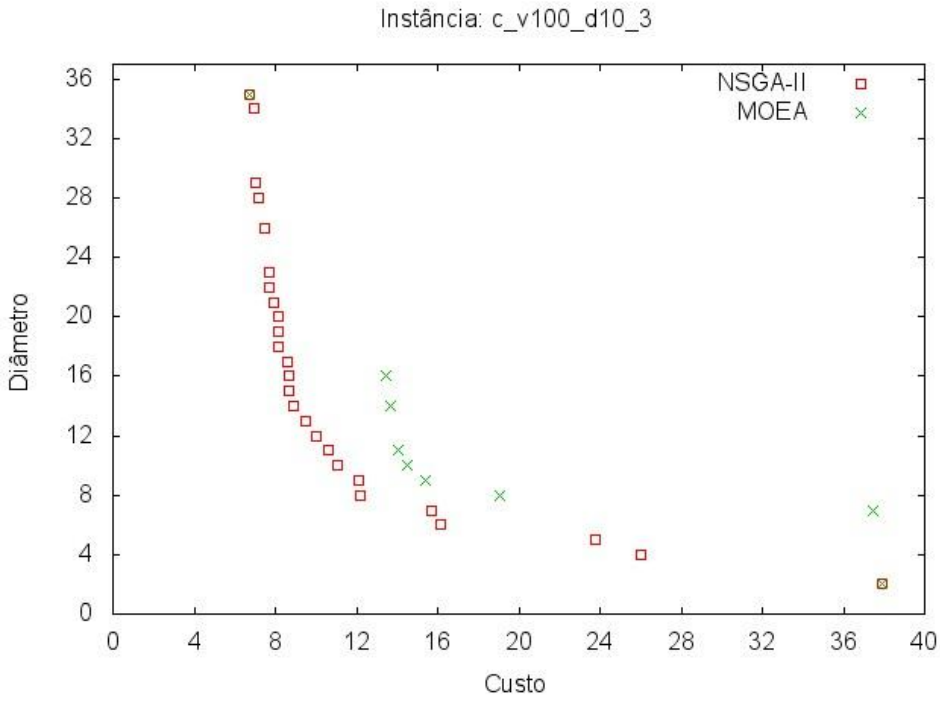


Figura 17 - Fronteira de Pareto encontrada pelo NSGA-II e MOEA para instância c_v100_d10_3

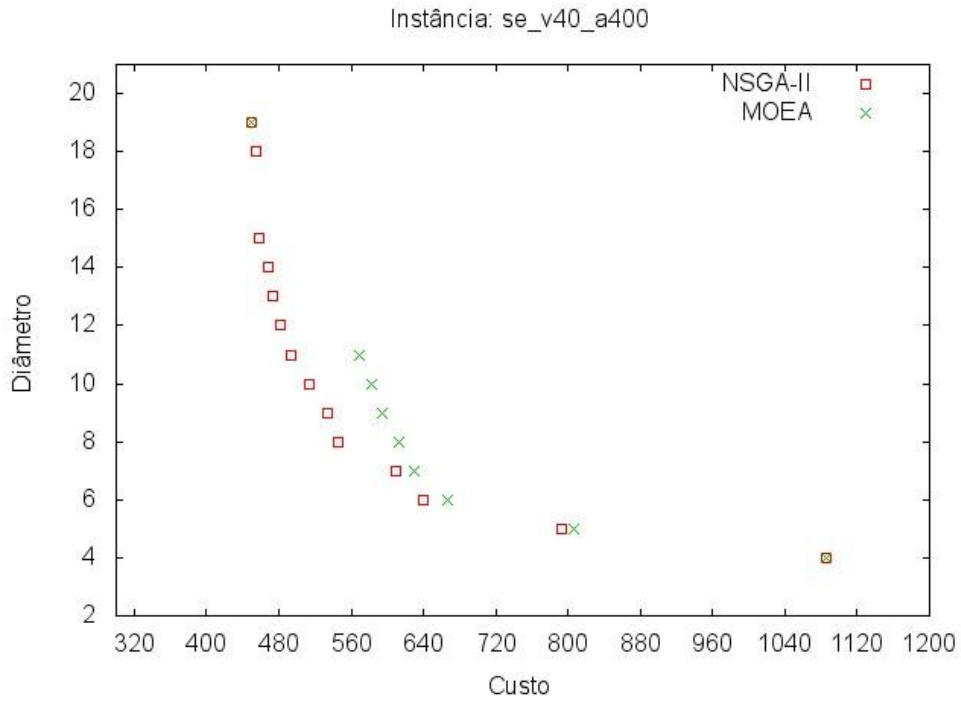


Figura 18 - Fronteira de Pareto encontrada pelo NSGA-II e MOEA para instância se_v40_a400

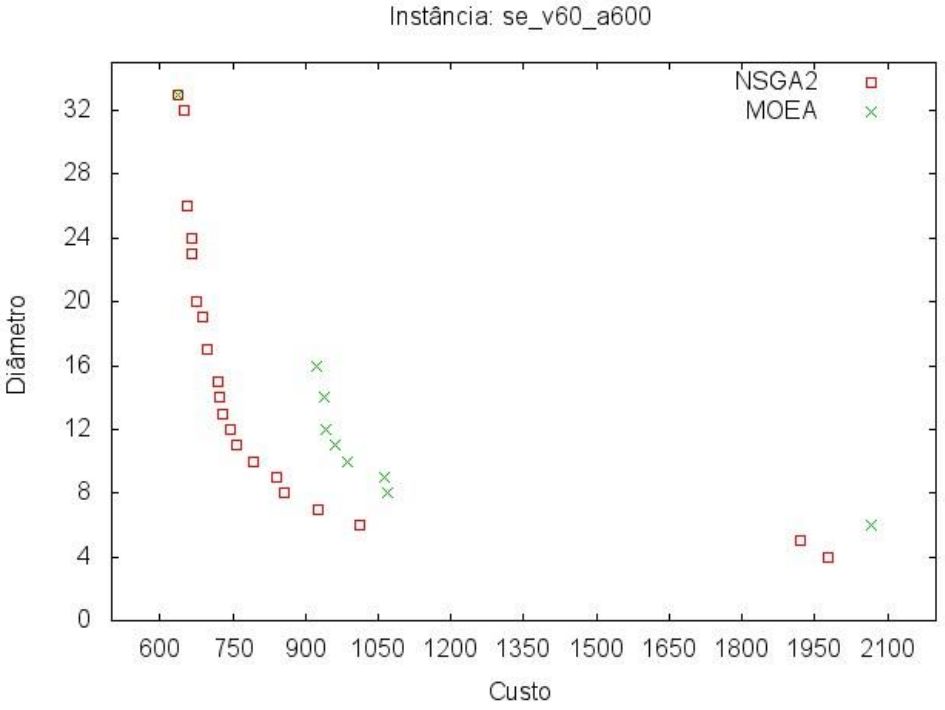


Figura 19 - Fronteira de Pareto encontrada pelo NSGA-II e MOEA para instância se_v60_a600

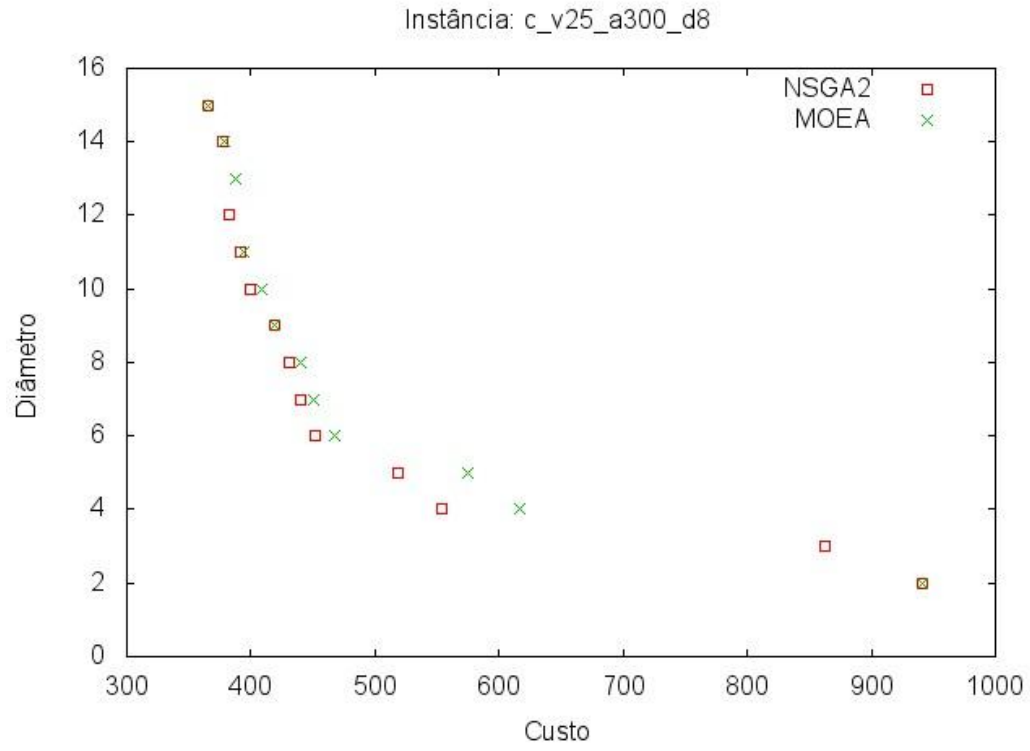


Figura 20 - Fronteira de Pareto encontrada pelo NSGA-II e MOEA para instância c_v25_a300_d8

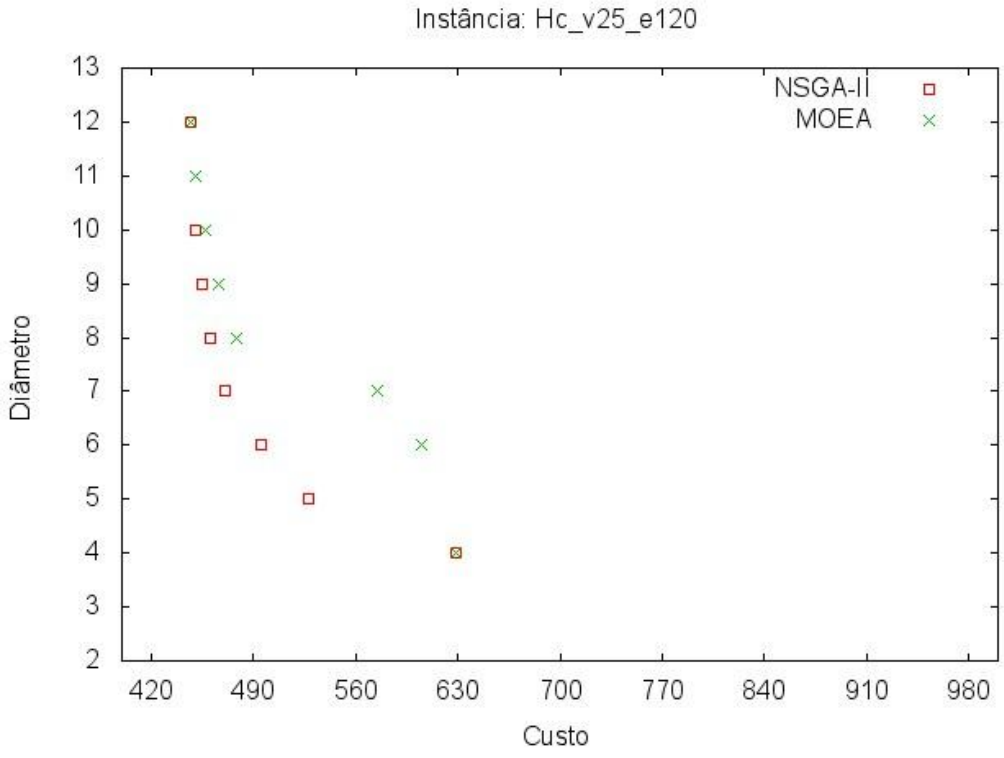


Figura 21 - Fronteira de Pareto encontrada pelo NSGA-II e MOEA para instância Hc_v25_e120

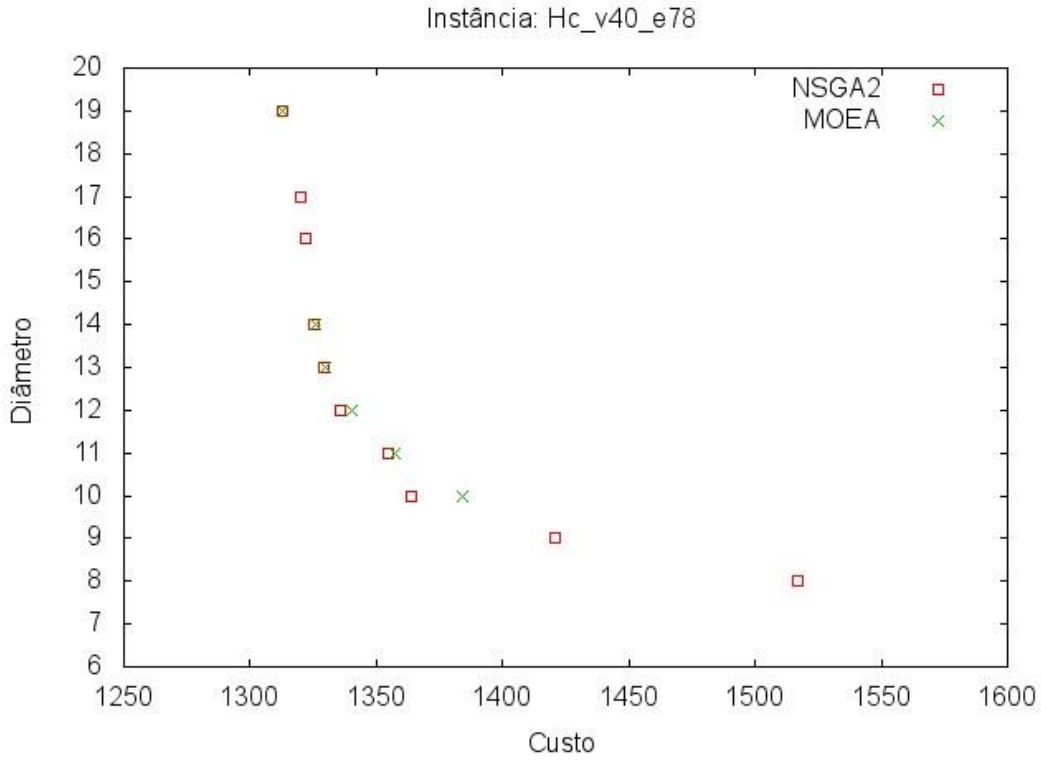


Figura 22 - Fronteira de Pareto encontrada pelo NSGA-II e MOEA para instância Hc_v40_e78

Nas figuras apresentadas tentamos mostrar os principais resultados obtidos após a execução de todas as instâncias dos quatro grupos. Resultados similares foram encontrados para todas as instâncias de cada um dos grupos. Pode-se observar que a fronteira encontrada pelo NSGA-II supera em todos os casos a fronteira obtida pelo MOEA, tanto na quantidade de soluções, como em qualidade das soluções e uniformidade da fronteira.

O MOEA possui um operador genético eficiente, mas sofre para encontrar soluções em diâmetros maiores. Isso provavelmente acontece porque ele não tem um operador genético que diversifique mais suas soluções. O MOEA para obter melhores soluções deveria inserir um método para guiar sua busca, tentando uniformizar a fronteira de Pareto.

O NSGA-II possui o operador *crowding distance*, que justamente tenta utilizar as soluções mais afastadas da fronteira da Pareto a fim de uniformizar a mesma. Isso faz com que os indivíduos escolhidos para a próxima iteração sejam aqueles que por estarem mais afastados precisam de mais soluções ao seu redor. Isso faz com que a fronteira do NSGA-II avance mais rapidamente produzindo sempre melhores soluções que os demais métodos.

7. CONCLUSÃO

No trabalho foi proposto um modelo exato para o problema da Árvore Geradora de Custo e Diâmetro Mínimos (bi-AGCDM). O modelo é baseado em multifluxo e em nosso conhecimento esta é a primeira modelagem dedicada a resolver o problema bi-AGCDM. Os resultados podem ser considerados satisfatórios e mostram a eficiência do modelo proposto, encontrando as melhores soluções para todas as instâncias testadas.

A utilização de uma metaheurística multiobjetivo para o bi-AGCDM também foi proposta neste trabalho. Utilizamos o NSGA-II aplicado ao problema. Este algoritmo foi escolhido por ser o mais referenciado para problemas bi-objetivo na literatura. Os métodos baseados em Algoritmos Genéticos são extremamente eficiente para problemas bi-obejtivos, devido a variedade e diversidade de seus operadores. Este algoritmo mostrou convergir rapidamente ao longo das iterações, conseguindo em todos os casos encontrar uma fronteira da Pareto com soluções eficientes.

O MOEA foi implementado por ser um algoritmo proposto recentemente e dedicado a resolver problemas bi-objetivo em árvores geradoras. A modificações foram feitas no algoritmo com o intuito de melhorar e deixar em iguais condições execução para se fazer uma comparação justa com o NSGA-II implementado.

Os resultados comprovaram e eficiência do NSGA-II em relação ao outro método, que apesar de eficiente, não consegue encontrar melhores soluções que o primeiro algoritmo.

O tema abordado no trabalho é recente e deixa uma série de pontos que podem ser trabalhados visando a melhoria dos modelos e algoritmos propostos. Novas estratégias e algoritmos e aprimoramento daqueles que já existem na literatura também pode ser abordado. A definição de outros operadores genéticos pode ser trabalhada na construção e evolução da população. Outras métricas de avaliação para problemas bi-objetivo podem ser aplicadas na verificação das soluções, com o *Spread* ou *General Distance* [VELDHUIZEN, D. A. V.; LAMONT, 2000].

Outro ponto de estudo seria nas diferentes formas de codificação de árvores, passíveis de serem utilizadas em algoritmos genéticos, com o objetivo de melhorar os tempos de execução de instâncias de grande porte. O desenvolvimento de outros métodos multiobjetivo heurísticos e exatos também podem ser tratados, desenvolvendo-se comparações entre as

diferentes estratégias utilizadas para resolver o problema de bi-AGCDM. Como o modelo geral desenvolvido nesta dissertação foi testado apenas para a versão mono-objetivo com o software CPLEX, pode-se cogitar também do uso de ferramentas como a "Global Optimization Toolbox" do MATLAB [PINTÉR *et al*, 2006], que é dedicada a resolver problemas multi-objetivo. Aplicações práticas reais dos algoritmos desenvolvidos também serão tratadas durante o doutorado.

Random-key genetic algorithms (RKGA) foram introduzidos por [BEAN, 1994] para a resolução de problemas de seqüenciamento em otimização combinatória. Desde então, eles têm sido estendidos para lidar com uma ampla classe de problemas de otimização combinatória [RESENDE, 2012; GONÇALVES e RESENDE, 2011; BURIOL *et al*, 2005]. Biased random-key genetic algorithms (BKRGA) são uma extensão dos RKGA que diferem apenas na maneira de escolher os indivíduos para o cruzamento. Enquanto no RKGA ambos os pais são selecionados da população elite, no BKRGA apenas um dos pais é escolhido desta forma, sendo o outro advindo da população não elite. Os algoritmos genéticos baseados em chaves aleatórias utilizam uma população de cromossomos que consistem em vetores de números reais com valores no intervalo [0,1]. Cada elemento do vetor é chamado de chave e seu valor é gerado aleatoriamente na população inicial. A adaptabilidade do cromossomo é definida pelo custo da solução fornecida por uma heurística que recebe como um de seus dados de entrada o vetor de chaves do cromossomo e devolve uma solução viável para o problema

Esta nova classe de algoritmos vem ganhando grande notoriedade na literatura, podendo-se assim trabalhar na extensão dos algoritmos baseados em chaves aleatórias para os problemas bi-critério ou bi-objetivo.

REFERÊNCIAS

ALNARENGA, F. V.; ROCHA, M. L. Uma Metaheurística Grasp Para o Problema da Árvore Geradora de Custo Mínimo com Grupamentos Utilizando Grafos Fuzzy. Journal of Computer Science, v. 5, p. 66 - 75, Março 2006.

ARAÚJO, A. P. F. BOERES,C. REBELLO, V. E.F, RIBEIRO C. C. A distributed and hierarchical strategy for autonomic grid-enabled cooperative metaheuristics with applications. International Transactions in Operational Research, 2011.

ARORA, S.; GARG, A. L. Clustering the data points to obtain optimum backbones for the Bounded Diameter Minimum Spanning Trees. International Conference on Communication Systems and Network Technologies, 2011.

BEAN, J. C.. Genetic algorithms and random Keys for sequencing and optimization. ORSA J. on Computing, 6:154 160, 1994

BINH, H. T. T.; HOAI, N. X.; MCKAY, R. I. A new hybrid Genetic Algorithm for solving the Bounded Diameter Minimum Spanning Tree problem. Congress on Evolutionary Computation. 2008. p. 3128 - 3134.

BINH, H. T. T.; MCKAY, R. I.; HOAI, N. X. New Heuristic and Hybrid Genetic Algorithm for Solving the Bounded Diameter Minimum Spanning Tree Problem. Genetic and Evolutionary Computation Conference. Montreal, Querbec, Canadá, July,2009.

BINH, H. T. T.; NGHIA, N. D. New Multi-parent Recombination in Genetic Algorithm for Solving Bounded Diameter Minimum Spanning Tree Problem. First Asian Conference on Intelligent Information and Database Systems. [S.l.]: [s.n.]. 2009.

BUI, M.; BUTELLE, F.; LAVAULT, C. A distributed algorithm for constructing the minimum spanning tree problem. Journal of Parallel and Distributed Computing, v. 64, p. 571 - 577, 2004.

BURIOL, L.S., RESENDE, M.G.C.,. RIBEIRO, C. C.; THORUP,, M. A hybrid genetic algorithm for the weight setting problem in OSPF/IS-IS routing. Networks, 4: 1, pp. 36-56, 2005

CARRANO, E. G., FONSECA, C. M., TAKAHASHI, R. H. C., PIMENTA, L. C. A. E NETO, O. M. A preliminary comparison of tree encoding schemes for evolutionary algorithms. p. 1969 –1974, Montreal, Canadá, 2007.

DEB, K. PRATAP,A. A. SAMEER AND T. MEYARIVAN. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, v. 6, April 2002. ISSN 2.

ERNST, A. T. A hybrid Lagrangian Particle Swarm Optimization Algorithm for the degree-constrained minimum spanning tree problem. Congress on Evolutionary Computation (CEC). 2010. p. 1 - 8.

FEOFILOFF, P.; KOHAYAKAWA, Y.; WAKABAYASHI, Y. Uma Introdução Sucinta à Teoria dos Grafos. USP, Julho 2011.

GAREY, M. R.; JOHNSON, D. S. Computers and intractability: A guide to the theory of NP-completeness. New York: W. H. Freeman and Company, 1979.

GONÇALVES, J.F.; RESENDE, M.G.C. Biased random-key genetic algorithms for combinatorial optimization. Journal of of Heuristics 17: 487-525, 2011.

GOUVEIA, L.; MAGNANTI, T. L. Network flow models for designing diameter-constrained minimum-spanning and steiner trees. Networks, v. 41, p. 159 - 173, 2003.

GOUVEIA, L.; SIMONETTI, L.; UCHOA, E. Modeling hop-constrained and diameter-constrained minimum spanning tree problems as steiner tree problems over layered graphs. Mathematical Programming, v. 128, p. 123 - 148, 2011.

GRUBER, M.; HEMERT, J. V.; RAIDL, G. R. Neighbourhood Searches for the Bounded Diameter Minimum Spanning Tree Problem Embedded in a VNS, EA, and ACO. Genetic and Evolutionary Computation Conference, Seattle, Washington, USA., July 2006.

GRUBER, M.; RAIDL, G. R. Exploiting Hierarchical Clustering for Finding Bounded Diameter Minimum Spanning Trees on Euclidean Instances. Genetic and Evolutionary Computation Conference, Montréal, Québec, Canada, July 2009a.

GRUBER, M.; RAIDL, G. R. Heuristic Cut Separation in a Branch&Cut Approach for the Bounded Diameter Minimum Spanning Tree Problem. International Symposium on Applications and the Internet. 2008. p. 261 - 264.

GRUBER, M.; RAIDL, G. R. Solving the Euclidean Bounded Diameter Minimum Spanning Tree Problem by Clustering–Based (Meta–) Heuristics. EUROCAST, v. 5717, p. 665 - 672, 2009b.

HASSIN, R.; TAMIR, A. On the minimum diameter spanning tree problem. Information Processing Letters, v. 53, p. 109 - 111, 1995.

HO, J. M. et al. Minimum diameter spanning trees and related problems. SIAM Journal on Computing, v. 20, p. 987 - 997, 1991.

JULSTROM, B. A. Greedy heuristics for the bounded diameter minimum spanning tree problem. ACM Journal of Experimental Algorithmics, v. 14, 2009.

JÚNIOR, J. B. R. VLACHOU, A., DOULKERIDIS, C AND NØRVÅG, K. Efficient Execution Plans for Distributed Skyline Query Processing. International Conference on Extending Database Technology, Uppsala, Sweden, p. 22 - 24, March 2011.

KUMAR, R.; BAL, B. K.; ROCKETT, P. I. Multiobjective Genetic Programming Approach to Evolving Heuristics for the Bounded Diameter Minimum Spanning Tree Problem. Genetic and Evolutionary Computation Conference, Montréal, Québec, Canada, July 2009.

KUMAR, R. AND SINGH, P. K. On quality performance of heuristic and evolutionary algorithms for biobjective minimum spanning trees. In GECCO '07: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, 2259, New York,m USA. ACM Press, 2007.

LIMA, T. W.; ROTHLAUF, F.; ALEXANDRE, C. B. The Node-Depth Encoding: Analysis and Application to the Bounded-Diameter Minimum Spanning Tree Problem. Genetic and Evolutionary Computation Conference, Atlanta, Georgia, USA., p. 12 - 16, July 2008.

LUCENA, A., RIBEIRO, C.; SANTOS, A. C. A hybrid heuristic for the diameter constrained minimum spanning tree problem. *Journal of Global Optimization*, v. 46, p. 363–381, 2010.

LUCENA, A.; RIBEIRO, C. C.; SANTOS, A. C. A hybrid heuristic for the diameter constrained minimum spanning tree problem. Journal of Global Optimization, v. 46, p. 363 - 381, 2010.

NEUMANN, F. AND WEGENER, I. Minimum spanning trees made easier via multi-objective optimization. Natural Computing, 5:305–319, 2006

NGHIA, N. D.; BINH, H. T. T. New Recombination Operator in Genetic Algorithm For Solving the Bounded Diameter Minimum Spanning Tree Problem. International Conference on Research, Innovation and Vision for the Future. 2007. p. 108 - 113.

NORONHA, T. F.; RIBEIRO, C. C.; SANTOS, A. C. Constraint programming for the diameter constrained minimum spanning tree problem. Electronic Notes in Discrete Mathematics, p. 93 - 98, 2008.

NORONHA, T. F.; RIBEIRO, C. C.; SANTOS, A. C. Solving diameter constrained minimum spanning tree problems by constraint programming. International Transactions in Operational Research, p. 653 - 665, 2010.

PATVARDHAN, C.; PRAKASH, V. P. Novel Deterministic Heuristics for Building Minimum Spanning Trees with Constrained Diameter. International Conference on Pattern Recognition and Machine Intelligence, v. 5909, p. 68 - 73, 2009.

PINTÉR, J. D.; LINDER, D.; CHIN, P. Global Optimization Toolbox for Maple: An introduction with illustrative applications. The Sixth International Conference on Optimization Techniques and Applications. Volume 21, p. 565-582, 2006.

RAIDL, G. R. ; JULSTROM, B. A. Greedy heuristics and an evolutionary algorithm for the bounded-diameter minimum spanning tree problem. *Proceedings of the 18th ACM Symposium on Applied Computing*, p. 747–752, Melbourne, USA, 2003b.

RAIDL, G. R.; JULSTROM, B. A. Edge sets: an effective evolutionary coding of spanning trees. *IEEE Transactions on Evolutionary Computation*, v. 7, n. 3, p. 225 – 239, 2003a.

RAQUEJO, C.; SANTOS, E. Greedy heuristics for the diameter-constrained Minimum spanning tree problem. Journal of Mathematical Sciences, v. 161, n. 6, 2009.

RESENDE, M. G. C.; RIBEIRO, C. C. Greedy randomized adaptive search procedures: Advances, hybridizations, and applications. Handbook of Metaheuristics, n. 2, p. 283 - 319, 2010.

RESENDE, M.G.C. Biased random-key genetic algorithms with applications in telecommunications. TOP, vol. 20:. 120-153, 2012

SAFARI, L.; RAHAMATI, A. Using Learning Automata to Solving Diameter Constrained Minimum Spanning Tree Problem. Australian Journal of Basic and Applied Sciences, p. 1265 - 1273, 2011.

SAHA, S. E KUMAR, R. Bounded-diameter MST instances with hybridization of multiobjective EA. *International Journal of Computer Applications*, v. 18, p. 17–25, 2011.

SAHA, S.; KUMAR, R. Improvement of Bounded-Diameter MST Instances with Hybridization of Multi-Objective EA. International Conference on Communication, Computing & Security, Rourkela, Odisha, India., p. 12 - 14, 2011.

SANTOS, A. C. Modelos e Algoritmos para o Problema da Árvore Geradora de Custo Mínimo com Restrição de Diâmetro. Tese de Doutorado, PUC-Rio, Rio de Janeiro, 2004.

SANTOS, A. C., LIMA, D. R.; ALOISE, D. J. Modeling the bi-objective diameter minimum spanning tree problem. *Proceedings of the Global Optimization Workshop (GOW)*, p. 125–128, Natal, Brasil, 2012.

SANTOS, A. C.; LUCENA, A.; RIBEIRO, C. C. Solving Diameter Constrained Minimum Spanning Tree Problems in Dense Graphs. Lecture Notes in Computer Science, v. 3059, p. 458 - 467, 2004.

VELDHUIZEN, D. A. V.; LAMONT, G. B. On measuring multiobjective evolutionary algorithm performance. *Proceedings of the Congress on Evolutionary Computation*, p. 204 – 211, California, USA, 2000.

W. THOMAS, H. CHEN, A. M. CAMPBEL . Network design for time-constrained delivery. Naval Research Logistics, 55: 493 – 515, 2008.

ZHOU, A., QU, B.-Y., LI, H., ZHAO, S.-Z., SUGANTHAN, P. N. E ZHANG, Q. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, v. 1, n. 1, p. 32 – 49, 2011.

ZITZLER, E., THIELE, L., LAUMANNS, M., FONSECA, C. E DA FONSECA, V. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, v. 7, n. 2, p. 117 – 132, 2003.