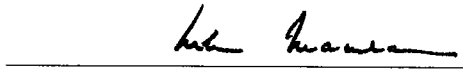
UM ALGORITMO POLINOMIAL PARA PROBLEMAS TRI-OBJETIVO DE OTIMIZAÇÃO DE CAMINHOS EM GRAFOS

Leizer de Lima Pinto

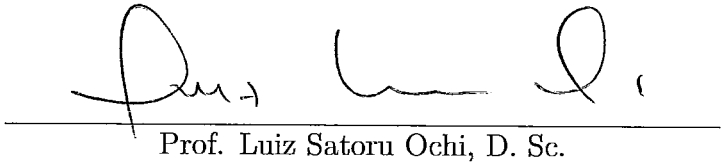
DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Prof. Cláudio Thomás Bornstein, D. Sc.



Prof. Nelson Maculan Filho, D. Sc.



Prof. Luiz Satoru Ochi, D. Sc.

RIO DE JANEIRO, RJ - BRASIL

ABRIL DE 2007

PINTO, LEIZER DE LIMA

Um Algoritmo Polinomial para Problemas Tri-Objetivo de Otimização de Caminhos em Grafos [Rio de Janeiro] 2007

VIII, 34 p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia de Sistemas e Computação, 2007)

Dissertação - Universidade Federal do Rio de Janeiro, COPPE

1. Otimização em Grafos
2. Programação Multi-Objetivo
3. Problema do Melhor Caminho Tri-Objetivo

I. COPPE/UFRJ II. Título (série)

Para José Benedito, Joaninha
e Lilian, pelo apoio de sempre.

Para Fellipe, por ter me moti-
vado para realizar este trabalho.

Agradecimentos

Apesar desta parte da tese ser opcional, eu não poderia deixar de agradecer algumas pessoas, pois sem elas este trabalho não teria se concretizado. Então, agradeço: ao professor Cláudio Bornstein, pela disposição em me orientar, pela atenção, pelos ensinamentos e por me proporcionar um ambiente de trabalho tão agradável, não somente durante esta tese, mas também no período em que desenvolvemos os softwares SPLINT e CamOtim; aos professores Maculan e Satoru, por aceitarem ao convite de participação na banca de defesa desta tese; ao professor Adilson Xavier, pela recepção na minha chegada ao PESC; ao professor Marco Antonio, da Universidade Católica de Goiás, por me introduzir ao mundo científico, pelas várias sugestões, por me preparar para os estudos aqui na COPPE e, também, pela amizade; às secretárias do PESC: Cláudia, Lúcia, Mercedes, Solange, Sônia, Patrícia, Carol e Taísa, por me esclarecerem várias dúvidas. Além da companhia agradável em algumas tardes no Grêmio da COPPE; ao amigo e irmão Elivelton, que não mediu esforços para ajudar no meu ingresso aqui no PESC. Além disso, me ensinou todos os detalhes para minha sobrevivência na cidade do Rio de Janeiro; aos parentes do Elivelton: Tia Isabel (101 anos), Glória, Guinez, Tetê e Zezé, que me receberam como se eu fosse um membro da família; aos colegas do LabOtim: Fátima, Yuri, Ádria, Luidi, Talita, Juliana, Sérgio, Rosa, Ronaldo, Pedro e Michele, que sempre estão prontos para me ajudar. Principalmente ao Yuri, pelas várias dicas em minhas implementações; ao amigão Vinícius (Parça), por proporcionar meu retorno ao futebol depois de tanto tempo, pela companhia agradável nas noites boêmias do Rio de Janeiro e por exercer tão bem o papel de amigo; à família do Parça: Tio Cláudio, Tia Sônia, patrão e Isabela, que me proporcionaram um natal (2006) maravilhoso; à Taísa, pela amizade e por me apresentar tantas pes-

soas legais, como a Paulista; aos 'família': Papel, Paulo Nunes, Dil, Targino e Parça, pelas tardes no mangue; aos colegas: Henrique, Ronaldo, Luiz Rigo, Sérgio Gonzalez, Watanabe, Fued, Leandro, Lasaro, Erika, Hebert, Daniel cabeção, Renato Caculé, André e Rodrigo, pela companhia; aos meus colegas com quem moro, Nilson e Florêncio, por me aturarem; ao meu pai José Benedito, à minha mãe Joaninha, à minha irmã Lilian e ao meu sobrinho Fellipe, que não medem esforços para me ajudar na realização dos meus objetivos. Pelo amor, pelo carinho, por compreenderem minha ausência em vários momentos importantes de suas vidas, e por me bancar (financeiramente) com tanta presteza, mesmo com as dificuldades, durante o período em que fiquei sem bolsa de estudos; à minha madrinha Ordalina, pela preocupação com o meu futuro e pelos presentes que eu jamais esquecerei, pois em certos momentos eles foram fundamentais para mim; aos meus amigos da minha querida cidade natal Itaguaru-GO, por compreenderem minha ausência; aos meus companheiros da época em que eu tocava boiada no sertão de Goiás, Dijalma (Abelha) e Sebastião (Bastião), que faleceram recentemente deixando muita saudade. E, finalmente, agradeço à CAPES pela bolsa de estudos concedida durante a maior parte do mestrado.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

UM ALGORITMO POLINOMIAL PARA PROBLEMAS TRI-OBJETIVO DE OTIMIZAÇÃO DE CAMINHOS EM GRAFOS

Leizer de Lima Pinto

Abril/2007

Orientador: Cláudio Thomás Bornstein

Programa: Engenharia de Sistemas e Computação

O foco deste trabalho é o problema de caminho ótimo tri-objetivo onde duas funções objetivo são do tipo gargalo, por exemplo, MinMax ou MaxMin. A terceira função objetivo pode ser do mesmo tipo ou podemos considerar uma função linear, por exemplo, MinSum ou MaxProd. Um algoritmo $O(m^2n^2)$ é apresentado para gerar o conjunto mínimo completo de soluções Pareto-ótimas, onde n e m são o número de nós e arcos do grafo. A geração do conjunto máximo completo também é possível. Demonstrações de otimalidade do algoritmo e extensões para vários casos especiais são apresentadas. Experimentos computacionais para problemas gerados aleatoriamente também são apresentados.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degreee of Master of Science (M.Sc.)

A POLYNOMIAL ALGORITHM FOR TRICRITERION PROBLEMS OF OPTIMIZATION OF PATHS IN GRAPHS

Leizer de Lima Pinto

April/2007

Advisor: Cláudio Thomás Bornstein

Department: Systems Engineering and Computer Science

The focus of this work is the tricriterion optimal path problem where two objective functions are of the bottleneck type, for example, MinMax or MaxMin. The third objective function may be of the same kind or we may consider a linear objective function like, for example, MinSum or MaxProd. An $O(m^2n^2)$ algorithm is presented that may generate the minimal complete set of Pareto-optimal solutions where n and m are the number of nodes and arcs of the graph. The computation of the maximal complete set is also possible. Optimality proofs are given and extensions for several other special cases are presented. Computational experience for randomly generated problems is reported.

Sumário

Capítulo 1

Introdução e Revisão Bibliográfica

O foco desta tese é o Problema do Melhor Caminho (PMC) em Grafos. Em particular, estamos interessados no PMC contendo três objetivos, onde pelo menos dois dos objetivos são do tipo "gargalo" (por exemplo o problema MaxMin). O outro objetivo pode ser, por exemplo, do tipo "soma" (por exemplo o problema MinSum), ou do tipo "produto" (por exemplo o problema MaxProd).

Tradicionalmente, o PMC contém apenas um objetivo, que consiste em minimizar a soma dos pesos dos arcos do caminho. Esse problema é mais conhecido na literatura como problema de caminho mínimo sendo aqui denominado de problema MinSum. Para a resolução deste problema temos, por exemplo, o algoritmo de Dijkstra [13] para pesos não-negativos e algoritmos de atualização de distâncias (veja, por exemplo, Bellman [4]). Em cada iteração do algoritmo de Dijkstra é obtido um caminho mínimo da origem até um outro nó do grafo. Assim em no máximo n iterações se obtém um caminho mínimo da origem até o destino, onde n é o número de nós do grafo. Já os algoritmos de atualização de distâncias melhoram as marcas dos nós (a marca de um nó representa o comprimento de um caminho da origem até o nó). Estes algoritmos são finalizados quando não é mais possível melhorar as marcas. Temos que o algoritmo de Dijkstra pertence

à classe dos algoritmos "caneta" (*label setting algorithms*) e os algoritmos de atualização de distâncias pertencem à classe dos algoritmos "lápis" (*label correcting algorithms*). Obviamente os algoritmos do tipo caneta são mais exigentes (exigem hipóteses mais fortes) do que os do tipo lápis. Outros PMC contendo um único objetivo e que podem ser resolvidos por variantes do algoritmo de Dijkstra e algoritmos de atualização de distâncias são:

1. MaxSum, que consiste em encontrar um caminho em que a soma dos pesos dos arcos seja máxima;
2. MinProd, que consiste em encontrar um caminho cujo produto dos pesos dos arcos seja mínimo;
3. MaxProd, que consiste em encontrar um caminho cujo produto dos pesos dos arcos seja máximo;
4. MinMax, que consiste em encontrar um caminho cujo maior peso dos arcos seja mínimo; e,
5. MaxMin, que consiste em encontrar um caminho cujo menor peso dos arcos seja máximo.

Estes problemas são freqüêntemente considerados na literatura de Otimização em Grafos (veja, por exemplo, Ahuja, Magnanti e Orlin [1], Gondran e Minoux [18] e Hansen [19]). Dois problemas que não podem ser resolvidos por algoritmos do tipo caneta, ou seja, não podem ser resolvidos por variantes do algoritmo de Dijkstra, são:

6. MinMin, que consiste em encontrar um caminho cujo menor peso dos arcos seja mínimo; e,
7. MaxMax, que consiste em encontrar um caminho cujo maior peso dos arcos seja máximo.

Para estes dois últimos problemas, Pinto e Bornstein [24] apresentam variantes de algoritmos de atualização de distâncias. Estes dois últimos problemas não serão mais considerados neste trabalho pela dificuldade em se encontrar aplicações práticas, ou seja, aparentemente não é fácil encontrar exemplos concretos onde estes objetivos se apliquem. Denominaremos como do tipo "gargalo" os objetivos 4, 5, 6 e 7.

Um software didático contendo implementações de variantes dos algoritmos de Dijkstra e atualização de distâncias para a resolução de todos esses problemas uni-objetivo citados acima, está disponível gratuitamente na internet (veja Pinto e Bornstein [25]).

Em muitas aplicações envolvendo o PMC, existe a necessidade de se considerar mais de um objetivo. Para uma leitura mais detalhada sobre aplicações do PMC multi-objetivo veja, por exemplo, Batta e Chiu [3], Current e Min [10] e Current e Marsh [12].

A meta principal da resolução de problemas com uma única função objetivo é a obtenção de uma solução ótima. Porém, quando o problema contém mais de uma função objetivo (problema multi-objetivo) a idéia de solução ótima é freqüentemente substituida pela idéia de solução Pareto-ótima (equivalentemente, solução eficiente ou solução não dominada). A meta desses problemas multi-objetivo é gerar um conjunto mínimo completo (ou máximo completo) de soluções Pareto-ótimas.

As primeiras definições de não dominância e solução Pareto-ótima surgiram em 1896 com Pareto [23]. O primeiro trabalho na área da Pesquisa Operacional (PO) utilizando esses conceitos de Pareto-ótimo foi apresentado por Koopmans [20] em 1951. Esses conceitos são fundamentais no campo da Programação Multi-Objetivo, que é estudada em diversas áreas da PO, por exemplo em Programação Linear (veja Zeleny [29]). A grande maioria das referências que iremos citar no decorrer deste trabalho, consideram problemas

multi-objetivo da área de Otimização em Grafos.

Martins [21] apresenta um algoritmo NP-difício do tipo caneta para a resolução do PMC multi-objetivo, onde todos os objetivos são MinSum. Para o mesmo problema, Guerriero e Musmanno [17] apresentam uma classe de métodos do tipo lápis. Uma revisão do algoritmo de Martins (citado acima) é apresentada por Gandibleux, Beugnies e Randriamasy [16], considerando, além dos multi-objetivo do tipo MinSum, um objetivo MaxMin. O PMC multi-objetivo também é estudado em Azevedo e Martins [2], Clímaco e Martins [7], Corley e Moon [9], Ehrgott e Gandibleux [14] e Tung e Chew [28].

No PMC bi-objetivo onde os dois objetivos são MinSum, o número de soluções Pareto-ótimas cresce exponencialmente para alguns grafos específicos (veja Hansen [19]). Então, no pior caso, os algoritmos para o PMC contendo mais de um objetivo MinSum, são de complexidade exponencial. As referências, Clímaco e Martins [8], Current, Revelle e Cohon [11] e Tung e Chew [27], também trabalham com o PMC com dois objetivos MinSum.

Para o PMC bi-objetivo composto por um objetivo MinSum e o outro MinMax, Hansen [19] apresenta um algoritmo de complexidade $O(m^2 \log n)$ e Berman, Einav e Handler [5] apresentam um algoritmo de complexidade $O(mn^2)$, onde n e m são, respectivamente, o número de nós e o número de arcos do grafo. Conseqüentemente o algoritmo de Hansen é menos eficiente do que o algoritmo de Berman, Einav e Handler para grafos densos e mais eficiente para grafos esparsos. Neste trabalho de Hansen são apresentados algoritmos para dez PMC bi-objetivo, e em Martins [22] são apresentados algoritmos para PMC bi-objetivo onde pelo menos um dos objetivos é do tipo gargalo.

O principal resultado deste trabalho consiste em um algoritmo de complexidade $O(m^2n^2)$ para o PMC tri-objetivo composto por um objetivo

MinSum, outro MinMax e o outro MaxMin. Como já dito anteriormente, procura-se a geração de soluções Pareto-ótimas. Denominaremos esse problema de MaxMin-MinMax-MinSum. O algoritmo considera ordenações nos pesos dos arcos associados aos problemas MaxMin e MinMax, e neste sentido, apresenta algumas semelhanças com o algoritmo de Berman, Einav e Handler.

O fato de aqui se considerar dois objetivos do tipo gargalo ao invés de um único, como em Berman, Einav e Handler [5], obriga a introdução de um teste adicional que compara algumas soluções já existentes com a nova solução gerada, verificando se houve melhoria do objetivo MinSum. Caso não haja melhoria, a nova solução é descartada.

O algoritmo trabalha com três tipos de pesos distintos para cada arco. Estes pesos são referentes, respectivamente, aos dois problemas de gargalo e ao problema MinSum. O algoritmo ordena os pesos dos arcos decrescentemente para o problema MaxMin e crescentemente para o problema MinMax. Para o conjunto assim ordenado, o algoritmo gera instâncias que correspondem a grafos onde se admitem, tão somente, arcos que respeitem os limites impostos pela ordenação. Para cada instância gera-se uma solução aplicando um algoritmo do tipo MinSum. O teste mencionado no parágrafo anterior verifica se esta é ou não um solução eficiente (Pareto-ótimo). Uma adaptação do algoritmo permite as opções de gerar um conjunto mínimo completo de soluções eficientes (uma solução para cada valor do vetor objetivo) ou então gerar o conjunto máximo completo (todas as soluções para cada valor do vetor objetivo).

No capítulo 2, apresentamos a definição do problema MaxMin-MinMax-MinSum, uma aplicação para este problema, e um algoritmo para a obtenção de um conjunto mínimo completo de soluções Pareto-ótimas. Além disso, é demonstrada a otimalidade do algoritmo e é feita uma modificação deste al-

goritmo para gerar o conjunto máximo completo. As implementações são desenvolvidas em linguagem de programação C. Também no capítulo 2, apresentamos os resultados computacionais para problemas gerados aleatoriamente com até 5000 nós.

Apesar do algoritmo ter sido desenvolvido para problemas de caminho em grafos, são possíveis extensões para o problema de árvore geradora ótima. Da mesma forma, embora, no algoritmo considerado o primeiro objetivo seja MaxMin e o segundo objetivo seja MinMax, nada impede que esta ordem seja trocada. Adicionalmente o terceiro objetivo (MinSum) pode ser substituído por MaxSum, MinProd ou MaxProd. Essas extensões são examinadas no capítulo 3.

No capítulo 3, definimos uma série de PMC tri-objetivo e adaptamos o algoritmo (denominado *Algoritmo MMS*) apresentado no capítulo 2 para esses problemas. Também apresentamos o problema MaxMin-MinMax-MinSum e variantes com restrições de gargalo, além de uma variante do *Algoritmo MMS* para o problema de árvore geradora MaxMin-MinMax-MinSum. Além disso, apresentamos um caso particular do problema MaxMin-MinMax-MinSum em que é possível reduzir o esforço computacional do *Algoritmo MMS*.

No capítulo 4 deste trabalho fazemos as considerações finais.

Capítulo 2

O Algoritmo

Nosso objetivo neste capítulo é definir o problema de caminho tri-objetivo MaxMin-MinMax-MinSum, apresentar uma aplicação para este problema, enunciar um algoritmo para resolvê-lo, demonstrar a otimalidade do algoritmo e apresentar resultados computacionais referentes a implementações do algoritmo.

2.1 O problema MaxMin-MinMax-MinSum

Considere o Grafo direcionado $G = (N, M)$, onde N é o conjunto dos nós e M o conjunto de arcos, com $|N| = n$ e $|M| = m$. Para cada $(i, j) \in M$ associamos as funções reais p^1, p^2 e p^3, ou seja, $p^1(i, j)$, $p^2(i, j)$ e $p^3(i, j) \in \Re$. Sejam m_1 e m_2 a quantidade de valores distintos assumidos pelas funções p^1 e p^2, respectivamente. Além disso, sejam $p^1_1, p^1_2, \ldots, p^1_{m_1}$ os diferentes valores assumidos por p^1 e $p^2_1, p^2_2, \ldots, p^2_{m_2}$ os diferentes valores assumidos por p^2. Sem perda de generalidade, suponhamos que:

$$p^1_1 > p^1_2 > \cdots > p^1_{m_1}, \tag{2.1}$$

$$p^2_1 < p^2_2 < \cdots < p^2_{m_2}. \tag{2.2}$$

Conforme será visto adiante, as ordenações feitas acima para p^1 e p^2

são fundamentais para a idéia do algoritmo apresentado neste capítulo. Para a função p^3 nenhuma ordenação será necessária.

Considere $M_{rg} = \{(i,j) \in M; p^1(i,j) \geq p_r^1 \text{ e } p^2(i,j) \leq p_g^2\}$. No decorrer deste trabalho usaremos o grafo parcial de G, $G_{rg} = (N, M_{rg})$. Evidentemente temos $M_{rg} \supseteq M_{r'g'}$ para $r' \leq r$ e $g' \leq g$. Basta ver que $M_{m_1 m_2} = M$.

Um *caminho* em G de origem $s \in N$ e destino $t \in N$ é um conjunto $P_{st} = \{s, i_1, i_2, \ldots, i_l, \ldots, i_h, t\} \subseteq N$ tal que i_l é o l-ésimo nó visitado. O conjunto de arcos do caminho P_{st} é dado por $AP_{st} = \{a_1, a_2, \ldots, a_{h+1}\} \subseteq M$, onde $a_1 = (s, i_1)$, $a_2 = (i_1, i_2)$, $\ldots$, $a_{h+1} = (i_h, t)$.

Considere S_G^{st} o conjunto de todos os caminhos de s a t em G. O problema que estamos interessados neste capítulo é o seguinte PMC tri-objetivo em Grafos:

$$
\begin{array}{lll}
(P) & \textit{maximizar} & \min\limits_{(i,j)\in AP_{st}} \{p^1(i,j)\} \\
 & \textit{minimizar} & \max\limits_{(i,j)\in AP_{st}} \{p^2(i,j)\} \\
 & \textit{minimizar} & \sum\limits_{(i,j)\in AP_{st}} p^3(i,j) \\
 & \textit{sujeito a:} & P_{st} \in S_G^{st}.
\end{array}
$$

Apresentamos a seguir algumas definições associadas ao problema (P) que utilizamos no decorrer deste capítulo.

Definições 2.1.1 *Considere o problema (P) e dois caminhos quaisquer de s a t em G, P_{st} e $\bar{P}_{st}$. Dizemos que:*

(a) $q^{P_{st}} = \left[q_1^{P_{st}}, q_2^{P_{st}}, q_3^{P_{st}}\right] \in \Re^3$, *onde*

$$q_1^{P_{st}} = \min_{(i,j)\in AP_{st}} \{p^1(i,j)\},$$

$$q_2^{P_{st}} = \max_{(i,j)\in AP_{st}} \{p^2(i,j)\},$$

$$e \quad q_3^{P_{st}} = \sum_{(i,j)\in AP_{st}} p^3(i,j),$$

é um ***vetor objetivo.***

(b) P_{st} ***domina*** $\bar{P}_{st}$ *se*

$$q_1^{P_{st}} \geq q_1^{\bar{P}_{st}}, \quad q_2^{P_{st}} \leq q_2^{\bar{P}_{st}} \quad e \quad q_3^{P_{st}} \leq q_3^{\bar{P}_{st}},$$

com pelo menos uma das desigualdades sendo atendida estritamente.

(c) P_{st} *é um* ***Pareto-ótimo*** *quando não existe outro caminho* $\hat{P}_{st}$ *de* s *a* t *tal que* $\hat{P}_{st}$ *domina* P_{st}.

(d) Um conjunto de soluções Pareto-ótimas distintas C^* *é um* ***conjunto mínimo completo*** *quando para qualquer Pareto-ótimo* P_{st} *uma das duas condições abaixo se verifica:*

(d_1*)* $P_{st} \in C^*$ *e não existe* $\hat{P}_{st} \in C^*$, $\hat{P}_{st} \neq P_{st}$ *tal que* $q^{\hat{P}_{st}} = q^{P_{st}}$.

(d_2*)* $P_{st} \notin C^*$ *e existe um único* $\hat{P}_{st} \in C^*$ *tal que* $q^{\hat{P}_{st}} = q^{P_{st}}$.

(e) O conjunto de todas as soluções Pareto-ótimas é o ***conjunto máximo completo.***

As definições de conjunto mínimo (e máximo) completo são feitas de maneira parecida em Gandibleux, Beugnies e Randriamasy [16]. A diferença é que eles também utilizam a definição de soluções equivalentes (Hansen [19]), que são Pareto-ótimos com o mesmo vetor objetivo.

Nosso principal objetivo neste capítulo é apresentar um algoritmo que gere um conjunto mínimo completo de Pareto-ótimos para o problema (P). Pode existir mais de um conjunto deste tipo.

Vejamos no próximo resultado que o número de elementos em um conjunto mínimo completo de Pareto-ótimos para (P) é polinomial.

Proposição 2.1.2 *Sejam n o número de nós do Grafo G e C^* um conjunto mínimo completo de Pareto-ótimos qualquer para o problema (P). Temos que $|C^*| \leq m_1 m_2 \leq m^2 \leq n^4$.*

Demonstração: Suponha, sem perda de generalidade, que existe algum caminho de s a t em G. Seja o caminho P_{st} um Pareto-ótimo qualquer para o problema (P). Obviamente, temos que $q^{P_{st}} = [p_r^1, p_g^2, q_3^{P_{st}}]$, para algum $r \in \{1, 2, \ldots, m_1\}$ e algum $g \in \{1, 2, \ldots, m_2\}$. Para qualquer outro caminho de s a t $\bar{P}_{st}$ tal que $q^{\bar{P}_{st}} = [p_r^1, p_g^2, q_3^{\bar{P}_{st}}]$, temos $q_3^{\bar{P}_{st}} \geq q_3^{P_{st}}$, pois P_{st} é um Pareto-ótimo. Deste modo, pela definição de conjunto mínimo completo, para cada par $i \in \{1, 2, \ldots, m_1\}$ e $j \in \{1, 2, \ldots, m_2\}$ temos no máximo um Pareto-ótimo em C^*. Como por outro lado temos $m_1 \leq m$, $m_2 \leq m$ e $m \leq n^2$, então, $|C^*| \leq m_1 m_2 \leq m^2 \leq n^4$. ∎

2.2 Uma aplicação

Considere um empresa que trabalha com o transporte terrestre de passageiros entre cidades. Para cada viagem que ela deve realizar de uma cidade s até outra cidade t, ela necessita decidir qual será o caminho, ou seja, quais os trechos percorridos. Cada arco (i, j) representa a estrada que vai da cidade i até a cidade j. Pressupomos que existe um único arco indo de i até j. Para cada arco existem três pesos associados, p^1, p^2 e p^3 tais que:

- $p^1(i, j)$ mede a qualidade da estrada representada pelo arco (i, j), quanto menor, pior a qualidade.
- $p^2(i, j)$ mede o congestionamento da estrada representada pelo arco (i, j), quanto maior, mais congestionada.
- $p^3(i, j)$ é o custo de percorrer a estrada representada pelo arco (i, j). Pode, por exemplo, ser proporcional à quilometragem percorrida.

O fato de uma estrada ser muito ruim pode implicar em quebras, perdas, atrasos e mesmo inviabilizar a viagem. O congestionamento pode,

por exemplo, causar muita insatisfação nos passageiros, além de aumentar o tempo da viagem. Além disso existem os custos com gasolina, pneus, óleo, etc. A empresa em questão pode estar interessada no problema de encontrar um caminho de s a t tal que:

- O pior trecho do caminho seja o melhor possível (MaxMin em p^1);
- O trecho mais congestionado do caminho tenha o menor congestionamento possível (MinMax em p^2); e,
- O custo pago para percorrer o caminho seja o menor possível (MinSum em p^3).

Portanto, como podemos observar, este é um problema do tipo (P).

2.3 Algoritmo MMS

Considerando o problema (P), vejamos a seguir um algoritmo para resolvê-lo. O vetor real $v = \{v_k\}$, que será usado pelo algoritmo e ainda não foi definido, serve como ferramenta auxiliar para identificar um novo Pareto-ótimo. Além disso, *AlgS* que também aparece no algoritmo, se refere a algum algoritmo para resolver o problema do caminho mínimo (MinSum). O grafo ao qual deverá ser aplicado o *AlgS* será G_{ij}, definido pelos valores específicos de i e j (veja *Algoritmo MMS*), utilizando-se exclusivamente os pesos p^3. Para efeito de *AlgS* os nós origem e destino serão sempre s e t, respectivamente. *AlgS* pode ser, por exemplo, o algoritmo de Dijkstra ou o algoritmo de atualização de distâncias. É bom lembrar que para o algoritmo de Dijkstra é necessário supor que todos os pesos dos arcos (considerando p^3) são não negativos. Para o algoritmo de atualização de distâncias basta supor a não existência de circuito com soma dos pesos dos arcos negativa. Segue-se o algoritmo.

Algoritmo 2.3.1 *MMS*

1. *Dados:* $G = (N, M)$, s, t
2. *Faça:* $C^* := \emptyset$, $Q_1 := \emptyset$, $Q_2 := \emptyset$, $Q_3 := \emptyset$ *e* $v_k := \infty, k = 1, 2, \ldots, m_2$
3. *Para* i *de* 1 *até* m_1 *Faça*
   - 3.1. *Para* j *de* 1 *até* m_2 *Faça*
     - 3.1.1. *Aplique AlgS em* G_{ij}
     - 3.1.2. *Seja* P_{st} *o caminho obtido. Se não existe caminho,* $q_3^{P_{st}} = \infty$
     - 3.1.3. *Se* $(q_3^{P_{st}} < \min_{1 \le k \le j} \{v_k\})$ *Então*
       - 3.1.3.1. $v_j := q_3^{P_{st}}$
       - 3.1.3.2. $C^* := C^* \bigcup \{P_{st}\}$
       - 3.1.3.3. $Q_1 := Q_1 \bigcup \{p_i^1\}$, $Q_2 := Q_2 \bigcup \{p_j^2\}$ *e* $Q_3 := Q_3 \bigcup \{q_3^{P_{st}}\}$
   - 3.2. *Fim Para* j
4. *Fim Para* i

É fácil ver que a complexidade do *Algoritmo MMS* é $O(m^2n^2)$. Temos que o laço externo é executado m_1 vezes, e para cada iteração deste laço, executamos m_2 vezes o laço interno. Logo o número total de iterações é m_1m_2. Como $m_1, m_2 \le m$, então no pior caso executamos m^2 iterações. O custo, no pior caso, de cada iteração é dado pela complexidade de *AlgS* (passo *3.1.1)*, que é $O(n^2)$ considerando o algoritmo de Dijkstra.

Vejamos no resultado que se segue que todos os caminhos inseridos em C^* pelo *Algoritmo MMS* são Pareto-ótimos para o problema (P), e que seus vetores objetivo são distintos.

Lema 2.3.2 *Seja* P_{st} *o caminho de* s *a* t *inserido em* C^* *na iteração* $i = r$ *e* $j = g$ *do Algoritmo MMS. Temos que:*

(a) $q_1^{P_{st}} = p_r^1$ *e* $q_2^{P_{st}} = p_g^2$; *e,*

(b) P_{st} *é um Pareto-ótimo para* (P).

Demonstração:

(a) Mostraremos, por contradição, que os valores p_r^1 e p_g^2 associados ao caminho P_{st} pelo *Algoritmo MMS*, são de fato os valores de $q_1^{P_{st}}$ e $q_2^{P_{st}}$. Como P_{st} foi obtido no grafo G_{rg}, então, $\forall (l,h) \in AP_{st}$ temos:

$$p^1(l,h) \geq p_r^1 \;\; e \;\; p^2(l,h) \leq p_g^2 \quad \Rightarrow \quad q_1^{P_{st}} \geq p_r^1 \;\; e \;\; q_2^{P_{st}} \leq p_g^2.$$

Agora suponhamos, sem perda de generalidade, que $q_1^{P_{st}} = p_{r'}^1 > p_r^1$ e $q_2^{P_{st}} = p_{g'}^2 \leq p_g^2$, onde $r' < r$ e $g' \leq g$. Logo, na iteração $i = r'$ e $j = g'$ anterior a $i = r$ e $j = g$, P_{st} é um caminho no grafo correspondente $G_{r'g'}$. Seja $\hat{P}_{st}$ a solução encontrada por $AlgS$ na iteração $i = r'$ e $j = g'$, no passo *3.1.2.* A otimalidade de $AlgS$ garante $q_3^{\hat{P}_{st}} \leq q_3^{P_{st}}$. Por outro lado, na iteração $i = r$ e $j = g$ o passo *3.1.3* garante que $q_3^{P_{st}} < \min\limits_{1 \leq k \leq g} \{v_k\} \leq \min\limits_{1 \leq k \leq g'} \{v_k\} \leq q_3^{\hat{P}_{st}}$. A demonstração da última desigualdade considera duas possibilidades:

(i) $\hat{P}_{st}$ não foi incorporado em C^* na iteração $i = r'$ e $j = g'$. Logo pelo passo *3.1.3* $\min\limits_{1 \leq k \leq g'} \{v_k\} \leq q_3^{\hat{P}_{st}}$ e, como os valores de v_k só diminuem, então está desigualdade continua verdade na iteração $i = r$ e $j = g$.

(ii) $\hat{P}_{st}$ foi incorporado em C^* na iteração $i = r'$ $j = g'$. Então pelo passo *3.1.3* $q_3^{\hat{P}_{st}} < \min\limits_{1 \leq k \leq g'} \{v_k\}$. No entanto no passo *3.1.3.1* fazemos $v_{g'} = q_3^{\hat{P}_{st}}$ obtendo assim $q_3^{\hat{P}_{st}} = \min\limits_{1 \leq k \leq g'} \{v_k\}$. Como os valores de v_k só diminuem, na iteração $i = r$ e $j = g$ temos $\min\limits_{1 \leq k \leq g'} \{v_k\} \leq q_3^{\hat{P}_{st}}$.

A contradição acima leva à negação do hipótese de que $q_1^{P_{st}} > p_r^1$. Logo $q_1^{P_{st}} = p_r^1$. Da mesma forma podemos demonstrar que $q_2^{P_{st}} = p_g^2$.

(b) Suponha, por contradição, que P_{st} não seja um Pareto-ótimo. Logo, por definição, existe uma solução $\hat{P}_{st}$ que domina P_{st}, ou seja,

$$q_1^{\hat{P}_{st}} \geq q_1^{P_{st}} = p_r^1, \;\; q_2^{\hat{P}_{st}} \leq q_2^{P_{st}} = p_g^2 \;\; e \;\; q_3^{\hat{P}_{st}} \leq q_3^{P_{st}}, \tag{2.3}$$

com pelo menos uma das desigualdades sendo atendida estritamente. As duas igualdades em (2.3) se devem ao item (a), já demonstrado. Suponhamos, sem perda de generalidade que $q_1^{\hat{P}_{st}} = p_{r'}^1 > q_1^{P_{st}} = p_r^1$ e $q_2^{\hat{P}_{st}} = p_{g'}^2 \leq q_2^{P_{st}} = p_g^2$ (a demonstração para os demais casos é semelhante). Logo $r' < r$, $g' \leq g$ e $\hat{P}_{st}$ é um caminho no grafo $G_{r'g'}$. Seja $\bar{P}_{st}$ a solução encontrada por $AlgS$ na iteração $i = r'$ e $j = g'$, que é anterior a $i = r$ e $j = g$. A otimalidade de $AlgS$ garante $q_3^{\bar{P}_{st}} \leq q_3^{\hat{P}_{st}}$. Por outro lado o passo *3.1.3*, aplicado na iteração $i = r$ e $j = g$

garante que $q_3^{P_{st}} < \min_{1 \leq k \leq g} \{v_k\} \leq \min_{1 \leq k \leq g'} \{v_k\} \leq q_3^{\bar{P}_{st}}$. A justificativa para as últimas duas desigualdades é idêntica a realizada para o item (a) da demonstração. Logo $q_3^{P_{st}} < q_3^{\bar{P}_{st}} \leq q_3^{\hat{P}_{st}}$, contrariando (2.3), o que contradiz a hipótese de que P_{st} não é solução Pareto-ótima.

∎

A parte (a) da demonstração se baseia no fato de que se $q_1^{P_{st}} > p_r^1$, então P_{st} será um caminho num grafo $G_{r'g'}$ gerado numa iteração anterior e portanto uma solução equivalente (que tem o vetor objetivo igual a $q^{P_{st}}$) ou melhor (que domina P_{st}) já teria sido incluída em C^*, barrando a inclusão de P_{st} na iteração $i = r$ e $j = g$.

A parte (b) da demonstração se baseia no fato de que se a solução P_{st} gerada pelo *Algoritmo MMS* não for Pareto-ótimo, existe uma solução $\hat{P}_{st}$ que a domina. Neste caso, $\hat{P}_{st}$ ou outra solução que também domina P_{st} já teria sido incluída em C^* numa iteração anterior. O passo *3.1.3* na iteração $i = r$ e $j = g$ teria impedido a inclusão de P_{st} em C^* o que é contrário à hipótese.

Agora, vejamos no próximo resultado que o nosso algoritmo gera um conjunto mínimo completo de Pareto-ótimos para (P).

Teorema 2.3.3 *No final do Algoritmo MMS temos que C^* é um conjunto mínimo completo de Pareto-ótimos para o problema (P).*

Demonstração: Pelo item (a) do *Lema 2.3.2* quaisquer duas soluções P_{st} e $\hat{P}_{st}$ inseridas pelo *Algoritmo MMS* em C^* são tais que $q^{P_{st}} \neq q^{\hat{P}_{st}}$. Logo temos satisfeita a condição (d_1) de *Definições 2.1.1* e garantimos que $P_{st} \neq \hat{P}_{st}$. Para satisfazer a definição de conjunto mínimo completo (d) basta verificar a condição (d_2), pois de (b) do *Lema 2.3.2* todas as soluções de C^* são Pareto-ótimos. Seja $P_{st} \notin C^*$ uma solução Pareto-ótima tal que $q_1^{P_{st}} = p_r^1$ e $q_2^{P_{st}} = p_g^2$. Logo, P_{st} é um caminho do grafo G_{rg}. Seja $\hat{P}_{st}$ o caminho obtido no passo *3.1.2* da iteração $i = r$ e $j = g$. A otimalidade de *AlgS* garante que $q_3^{\hat{P}_{st}} \leq q_3^{P_{st}}$. Além disso, como $\hat{P}_{st}$ foi obtido no grafo G_{rg}, então, por

definição $\forall (l,h) \in A\hat{P}_{st}$ temos $p^1(l,h) \geq p^1_r$ e $p^2(l,h) \leq p^2_g$, o que implica em $q_1^{\hat{P}_{st}} \geq p^1_r = q_1^{P_{st}}$ e $q_2^{\hat{P}_{st}} \leq p^2_g = q_2^{P_{st}}$. Juntando todos os resultados acima temos portanto $q_1^{\hat{P}_{st}} \geq q_1^{P_{st}}$, $q_2^{\hat{P}_{st}} \leq q_2^{P_{st}}$ e $q_3^{\hat{P}_{st}} \leq q_3^{P_{st}}$. Como P_{st} é um Pareto-ótimo, então nenhuma destas três últimas desigualdades pode ser estrita, pois senão P_{st} seria dominado por $\hat{P}_{st}$. Logo $q^{P_{st}} = q^{\hat{P}_{st}}$ e, portanto, $\hat{P}_{st}$ também é um Pareto-ótimo.
Mostraremos, a seguir, que no passo *3.1.3* da iteração $i = r$ e $j = g$ temos que ter, necessariamente, $q_3^{\hat{P}_{st}} < \min\limits_{1 \leq k \leq g} \{v_k\}$, o que nos garante $\hat{P}_{st} \in C^*$. Isto será mostrado por contradição.
Suponha, por contradição, que $q_3^{\hat{P}_{st}} \geq \min\limits_{1 \leq k \leq g} \{v_k\}$. Então existe um caminho $\bar{P}_{st}$ tal que $q_3^{\bar{P}_{st}} \leq q_3^{\hat{P}_{st}}$ inserido em C^* numa iteração $i = r'$ e $j = g'$ tal que $r' \leq r$ e $g' \leq g$, com pelo menos uma das desigualdades sendo atendida estritamente. Por (a) do *Lema 2.3.2* temos que $q_1^{\bar{P}_{st}} = p^1_{r'} \geq p^1_r = q_1^{\hat{P}_{st}}$ e $q_2^{\bar{P}_{st}} = p^2_{g'} \leq p^2_g = q_2^{\hat{P}_{st}}$, com pelo menos uma desigualdade sendo atendida de forma estrita. Logo $\bar{P}_{st}$ domina $\hat{P}_{st}$, ou seja, $\hat{P}_{st}$ não é um Pareto-ótimo, o que leva a um absurdo. Portanto, sendo satisfeito o teste do passo *3.1.3* temos $\hat{P}_{st} \in C^*$, o que satisfaz a condição (d_2) de conjunto mínimo completo.

∎

2.4 Conjunto máximo completo

Nesta seção vamos apresentar uma variante do *Algoritmo MMS* para gerar todas as soluções Pareto-ótimas para o problema (P), ou seja, o conjunto máximo completo de soluções Pareto-ótimas. É importante lembrar que o número de soluções deste conjunto pode crescer exponencialmente com o número de nós do grafo (veja, por exemplo, Ehrgott e Gandibleux [14]).

Seja $i = r$ e $j = g$ uma iteração do *Algoritmo MMS* em que inserimos P_{st} em C^*. Se nesta iteração tivéssemos gerado todos os caminhos ótimos de s a t contidos no grafo G_{rg}, ao invés de apenas P_{st}, então teríamos obtido todas as soluções Pareto-ótimas com vetor objetivo igual a $q^{P_{st}}$. Ou seja, para gerar o conjunto máximo completo de soluções Pareto-ótimas, basta substituir *AlgS* no passo *3.1.1* do *Algoritmo MMS* por um algoritmo que

gera todos os caminhos ótimos de s a t. Obviamente que no passo *3.1.3.2* devemos inserir em C^* todos esses caminhos obtidos no novo passo *3.1.1.*

A seguir apresentamos uma variante do algoritmo de Dijkstra para o problema de encontrar todos os caminhos ótimos de s a t em G_{rg}. Em Walczak e Wojciechowski [30], mostra-se como obter esta variante. Lembramos que este problema também pode ser resolvido por uma variante do algoritmo de atualização de distâncias. Para a variante que vamos apresentar é necessário supor que todos os pesos (p^3) dos arcos são positivos. Antes de apresentar o algoritmo, vejamos algumas notações que estaremos usando.

Notações:

F : Conjunto de nós 'fechados', ou seja, nós para os quais já encontramos todos os caminhos mínimos partindo do nó origem s.

f : Último nó fechado pelo algoritmo.

o_j : Soma dos pesos dos arcos de algum caminho de s a j.

c_j : Quantidade de caminhos de s a j com custo o_j.

C_j^k : k-ésimo caminho de s a t com custo o_j.

Algoritmo 2.4.1 S^T

1. *Dados:* $G_{rg} = (N, M_{rg})$, s, t
2. *Faça:* $f := s$, $F := \{s\}$, $C_s^1 := \{s\}$, $c_j := 0$ *e* $o_j := \infty$, $\forall j \in N - \{s\}$
   $c_s := 1$, $o_s := 0$
3. *Repita*
   - 3.1. *Se* $(f = t)$ *Então*
     - 3.1.1. $C_t^1, C_t^2, \ldots, C_t^{c_t}$ *são os caminhos ótimos de s a t*
     - 3.1.2. o_t *é o valor ótimo associado aos caminhos encontrados*
     - 3.1.2. *PARAR ALGORITMO*

3.2. Senão

3.2.1. Para todo $(f,j) \in M_{rg}$ *tal que* $j \notin F$ *Faça*

3.2.1.1. Se $(o_j > o_f + p^3(f,j))$ *Então*

3.2.1.1.1. $o_j := o_f + p^3(f,j)$

3.2.1.1.2. Delete $C_j^1, C_j^2, \ldots, C_j^{c_j}$

3.2.1.1.3. $c_j := c_f$

3.2.1.1.4. Para k *de* 1 *até* c_f *Faça*

3.2.1.1.4.1. $C_j^k := C_f^k \bigcup \{j\}$, *onde* j *é o último elemento de* C_j^k

3.2.1.2. Senão se $(o_j = o_f + p^3(f,j))$ *Então*

3.2.1.2.1. Para k *de* $c_j + 1$ *até* $c_j + c_f$ *Faça*

3.2.1.2.1.1. $C_j^k := C_f^k \bigcup \{j\}$, *onde* j *é o último elemento de* C_j^k

3.2.1.2.2. $c_j := c_j + c_f$

3.2.2. Fim Para (3.2.1)

3.2.3. Seja $h \notin F$ *tal que* $o_h \leq o_j, \forall j \notin F$

3.2.4. Se $(o_h = \infty)$ *Então*

3.2.4.1. Não existe caminho de s *a* t *em* G_{rg}

3.2.4.2. PARAR ALGORITMO

3.2.5. Senão

3.2.5.1. $F := F \bigcup \{h\}$

3.2.5.2. $f := h$

4. Fim Repita

Portanto trocando *AlgS* pelo *Algoritmo* S^T e inserindo em C^* todos os caminhos obtidos no novo passo *3.1.3.2* do *Algoritmo MMS*, temos um algoritmo para gerar o conjunto máximo completo de soluções Pareto-ótimas.

2.5 Implementações

Nesta seção vamos apresentar resultados computacionais referentes as implementações do *Algoritmo MMS*, para a obtenção de um conjunto mínimo

completo, e de sua variante apresentada na seção anterior, para a obtenção do conjunto máximo completo de soluções Pareto-ótimas.

Nossas implementações foram desenvolvidas na linguagem de programação C utilizando um computador pessoal com processador DURON 997 MHz e memória RAM de 128 Mb. Para resolver problemas com 50 e 1000 nós foi utilizado este mesmo computador. Para problemas com 5000 nós foi utilizado um computador pessoal com processador CENTRINO CORE DUO 1.6 GHz e 1024 Mb de memória RAM.

Os problemas que resolvemos para testar nossas implementações foram gerados aleatoriamente. Consideramos grafos com 50, 1000 e 5000 nós. Dividimos os problemas em três classes, que são:

***Prob*$_1$.** Para cada par de nós distintos i e j é gerado um número inteiro aleatório entre 1 e 10. (i, j) só será um arco do grafo se este número for 1, ou seja, existe uma probabilidade de $0,1$ de haver o arco (i, j).

***Prob*$_2$.** Para cada par de nós distintos i e j é gerado um número aleatório que pode ser 0 ou 1. (i, j) só será um arco do grafo se este número for 1, ou seja, existe uma probabilidade de $0,5$ de haver o arco (i, j).

***Prob*$_3$.** Para cada par de nós distintos i e j, temos que (i, j) é um arco do grafo, ou seja, são grafos completos.

Para todos os problemas, geramos aleatoriamente três pesos para cada arco. Conforme será visto nas tabelas de resultados, limitamos a quantidade de pesos distintos para as funções p^1 e p^2. Para estas funções geramos números inteiros aleatórios entre 1 e um dado valor, que no máximo é 70. Para p^3 geramos números inteiros aleatórios entre 1 e 10000. Para $AlgS$ implementamos o algoritmo de Dijkstra.

Consideremos *cmc* e *cMc* como sendo, respectivamente, o número de elementos dos conjuntos mínimo e máximo completo de Pareto-ótimos, referentes aos problemas que serão resolvidos pelas nossas implementações. Nas colunas $tempo_{cmc}$ e $tempo_{cMc}$ das tabelas apresentamos, respectivamente, o tempo (em segundos) gasto para a obtenção dos conjuntos mínimo e máximo completo. Além disso, lembramos que m, m_1 e m_2 são, respectivamente, o número de arcos e a quantidade de pesos distintos para as funções p^1 e p^2.

Conclusões sobre os resultados computacionais serão apresentados somente no capítulo 4.

Nas tabelas 2.1, 2.2 e 2.3 consideramos, respectivamente, grafos com 50, 1000 e 5000 nós. Seguem-se as tabelas.

$classe$	m	m_1	m_2	cmc	$tempo_{cmc}$	cMc	$tempo_{cMc}$
$Prob_1$	264	5	5	1	0,000	1	0,000
$Prob_1$	242	5	5	8	0,000	8	0,010
$Prob_1$	239	5	5	2	0,000	2	0,000
$Prob_1$	255	10	5	4	0,000	4	0,000
$Prob_1$	254	10	5	1	0,000	1	0,000
$Prob_1$	259	10	5	11	0,000	11	0,000
$Prob_1$	260	20	49	10	0,020	10	0,020
$Prob_1$	254	20	50	12	0,020	12	0,030
$Prob_1$	237	20	50	20	0,030	20	0,030
$Prob_1$	272	70	70	25	0,200	25	0,200
$Prob_1$	247	68	69	19	0,110	19	0,110
$Prob_1$	251	67	66	21	0,160	21	0,160
$Prob_2$	1249	5	5	11	0,000	11	0,010
$Prob_2$	1233	5	5	7	0,000	7	0,000
$Prob_2$	1193	5	5	18	0,000	19	0,000
$Prob_2$	1187	10	5	15	0,000	15	0,010
$Prob_2$	1240	10	5	23	0,000	23	0,000
$Prob_2$	1251	10	5	17	0,000	17	0,010
$Prob_2$	1203	20	50	20	0,050	20	0,060
$Prob_2$	1190	20	50	53	0,060	53	0,060
$Prob_2$	1220	20	50	17	0,050	17	0,060
$Prob_2$	1185	70	70	59	0,340	59	0,350
$Prob_2$	1228	70	70	13	0,140	13	0,130
$Prob_2$	1208	70	70	59	0,220	59	0,230
$Prob_3$	2450	5	5	13	0,000	13	0,000
$Prob_3$	2450	5	5	6	0,000	6	0,000
$Prob_3$	2450	5	5	16	0,000	16	0,010
$Prob_3$	2450	10	5	15	0,000	15	0,010
$Prob_3$	2450	10	5	22	0,000	22	0,010
$Prob_3$	2450	10	5	18	0,000	18	0,010
$Prob_3$	2450	20	50	65	0,070	65	0,060
$Prob_3$	2450	20	50	64	0,090	64	0,090
$Prob_3$	2450	20	50	30	0,070	30	0,070
$Prob_3$	2450	70	70	66	0,270	66	0,280
$Prob_3$	2450	70	70	30	0,300	30	0,300
$Prob_3$	2450	70	70	88	0,320	88	0,320

Tabela 2.1: *número de Pareto-ótimos e tempo de execução, grafos com 50 nós.*

$classe$	m	m_1	m_2	cmc	$tempo_{cmc}$	cMc	$tempo_{cMc}$
$Prob_1$	99801	5	5	19	0,952	20	0,991
$Prob_1$	99865	5	5	13	0,371	13	0,420
$Prob_1$	99802	5	5	17	0,791	17	0,852
$Prob_1$	100160	10	5	36	2,243	36	2,324
$Prob_1$	99743	10	5	33	2,273	33	2,354
$Prob_1$	99462	10	5	22	0,851	22	0,952
$Prob_1$	100061	20	50	109	18,867	109	20,079
$Prob_1$	99732	20	50	133	24,596	133	26,147
$Prob_1$	99584	20	50	192	26,959	192	28,581
$Prob_1$	100223	70	70	294	111,501	294	118,710
$Prob_1$	99703	70	70	216	76,110	216	81,246
$Prob_1$	99307	70	70	280	111,440	280	117,780
$Prob_2$	499438	5	5	17	0,901	17	0,952
$Prob_2$	499600	5	5	18	0,891	18	0,932
$Prob_2$	499352	5	5	21	1,031	22	1,112
$Prob_2$	499299	10	5	38	1,923	38	2,043
$Prob_2$	499644	10	5	34	2,013	35	2,133
$Prob_2$	499643	10	5	30	2,113	30	2,213
$Prob_2$	499422	20	50	120	19,398	120	21,150
$Prob_2$	499485	20	50	207	42,571	208	44,895
$Prob_2$	499603	20	50	189	45,826	189	48,309
$Prob_2$	499599	70	70	326	302,245	326	317,967
$Prob_2$	499378	70	70	352	140,482	352	151,588
$Prob_2$	499518	70	70	311	176,023	312	188,281
$Prob_3$	999000	5	5	23	1,533	23	1,682
$Prob_3$	999000	5	5	18	1,172	18	1,242
$Prob_3$	999000	5	5	24	1,502	24	1,642
$Prob_3$	999000	10	5	33	3,175	33	3,394
$Prob_3$	999000	10	5	37	1,883	37	2,003
$Prob_3$	999000	10	5	26	2,333	26	2,564
$Prob_3$	999000	20	50	207	40,438	207	44,114
$Prob_3$	999000	20	50	177	46,537	177	50,823
$Prob_3$	999000	20	50	211	44,023	212	47,839
$Prob_3$	999000	70	70	319	134,443	319	148,544
$Prob_3$	999000	70	70	345	264,611	346	285,280
$Prob_3$	999000	70	70	303	176,474	305	190,233

Tabela 2.2: *número de Pareto-ótimos e tempo de execução, grafos com 1000 nós.*

$classe$	m	m_1	m_2	cmc	$tempo_{cmc}$	cMc	$tempo_{cMc}$
$Prob_1$	2499486	5	5	20	3,297	20	3,375
$Prob_1$	2501021	5	5	23	5,297	23	5,359
$Prob_1$	2498509	5	5	21	7,516	21	7,578
$Prob_1$	2499774	10	5	30	13,250	30	13,421
$Prob_1$	2502192	10	5	35	14,531	35	14,750
$Prob_1$	2501547	10	5	32	9,921	32	10,063
$Prob_1$	2499052	20	50	196	192,032	197	194,937
$Prob_1$	2501360	20	50	221	208,094	221	210,109
$Prob_1$	2498822	20	50	239	202,984	239	206,203
$Prob_1$	2501006	70	70	383	1045,500	383	1050,500
$Prob_1$	2499927	70	70	415	950,188	415	950,859
$Prob_1$	2501244	70	70	383	683,110	383	704,140
$Prob_2$	12497492	5	5	19	5,907	21	6,156
$Prob_2$	12497671	5	5	24	9,032	24	9,328
$Prob_2$	12497681	5	5	17	5,453	17	5,578
$Prob_2$	12497391	10	5	31	16,015	33	16,813
$Prob_2$	12497377	10	5	27	8,578	28	8,969
$Prob_2$	12497336	10	5	38	15,985	39	16,593
$Prob_2$	12497455	20	50	246	318,890	247	328,985
$Prob_2$	12497473	20	50	238	244,234	240	251,797
$Prob_2$	12497502	20	50	253	293,344	256	302,844
$Prob_2$	12497598	70	70	399	998,141	400	1035,312
$Prob_2$	12497657	70	70	437	1286,734	437	1338,437
$Prob_2$	12497558	70	70	463	1455,215	467	1472,359
$Prob_3$	24995000	5	5	21	7,063	21	7,656
$Prob_3$	24995000	5	5	18	8,454	21	9,093
$Prob_3$	24995000	5	5	21	5,500	22	5,922
$Prob_3$	24995000	10	5	33	14,219	37	15,046
$Prob_3$	24995000	10	5	39	15,375	41	16,407
$Prob_3$	24995000	10	5	38	14,921	40	16,047
$Prob_3$	24995000	20	50	252	317,609	256	334,422
$Prob_3$	24995000	20	50	212	366,718	216	388,110
$Prob_3$	24995000	20	50	195	253,594	198	269,422
$Prob_3$	24995000	70	70	451	1782,250	454	1882,219
$Prob_3$	24995000	70	70	468	1700,860	472	1786,234
$Prob_3$	24995000	70	70	446	1105,609	447	1181,344

Tabela 2.3: *número de Pareto-ótimos e tempo de execução, grafos com 5000 nós.*

Capítulo 3

Extensões do Algoritmo MMS

Nosso objetivo neste capítulo é apresentar algumas extensões do *Algoritmo MMS* apresentado no capítulo anterior. Neste sentido, apresentamos primeiramente, em **3.1**, variantes deste algoritmo para uma classe de PMC tri-objetivo. Em **3.2** apresentamos o problema MaxMin-MinMax-MinSum com restrições e, em **3.3**, examinamos problemas de árvore geradora tri-objetivo. Na seção **3.4** apresentamos um caso particular do problema MaxMin-MinMax-MinSum em que é possível reduzir o esforço computacional do *Algoritmo MMS*.

Assim como no capítulo 2, considere $G = (N, M)$ um grafo direcionado, P_{st} um caminho de s a t neste grafo, AP_{st} o conjunto de arcos de P_{st}, S_G^{st} o conjunto de todos os caminhos de s a t em G, e p^1, p^2 e p^3 funções reais associadas aos arcos de G, ou seja, $p^1(i,j)$, $p^2(i,j)$ e $p^3(i,j) \in \Re$, $\forall (i,j) \in M$.

3.1 Variantes de PMC tri-objetivo

Nesta seção vamos definir diversos PMC tri-objetivo que podem ser resolvidos por variantes do *Algoritmo MMS*. Para isso, vamos primeiramente, definir alguns dos problemas uni-objetivo citados na introdução deste trabalho e que estaremos considerando nesta seção. Seguem-se as definições.

$$MaxSum: \quad \max_{P_{st} \in S_G^{st}} \left\{ \sum_{(i,j) \in AP_{st}} \{p^3(i,j)\} \right\}.$$

$$MinProd: \quad \min_{P_{st} \in S_G^{st}} \left\{ \prod_{(i,j) \in AP_{st}} \{p^3(i,j)\} \right\}.$$

$$MaxProd: \quad \max_{P_{st} \in S_G^{st}} \left\{ \prod_{(i,j) \in AP_{st}} \{p^3(i,j)\} \right\}.$$

$$MinMax: \quad \min_{P_{st} \in S_G^{st}} \left\{ \max_{(i,j) \in AP_{st}} \{p^3(i,j)\} \right\}.$$

$$MaxMin: \quad \max_{P_{st} \in S_G^{st}} \left\{ \min_{(i,j) \in AP_{st}} \{p^3(i,j)\} \right\}.$$

Considere (P_3) como sendo qualquer um dos cinco problemas definidos acima e, $AlgP_3$ um algoritmo para resolver (P_3).

Como já foi dito no capítulo 1, todos estes problemas podem ser resolvidos por variantes do algoritmo de Dijkstra ou por algoritmos de atualização de distâncias. No que tange Dijkstra, para os problemas MinSum, MaxSum, MinProd e MaxProd, é necessário fazer alguma hipótese sobre os pesos dos arcos. Para os problemas MinMax e MaxMin nenhuma hipótese nos pesos dos arcos é necessária. No que tange o algoritmo de atualização de distâncias, basta evitar circuitos patológicos (para o problema MaxSum, por exemplo, circuito com soma dos pesos dos arcos positiva).

Substituindo *AlgS* por $AlgP_3$ no *Algoritmo MMS*, podemos resolver o PMC tri-objetivo MaxMin-MinMax-(P_3). Lembramos que o teste realizado no passo *3.1.3* do *Algoritmo MMS* é para verificar se houve melhoria, com relação a algumas soluções já obtidas. Para o caso de maximização (MaxSum, MaxProd e MaxMin), no passo *3.1.3* o sinal de menor deve ser substituído

pelo sinal de maior e a função min pela função max. Além disso, no passo *2* devemos ter $v_k := -\infty$ e no passo *3.1.2* $q_3^{P_{st}} = -\infty$. Para o caso de minimização (MinSum, MinProd e MinMax) os passos *2*, *3.1.2* e *3.1.3* são iguais aos do *Algoritmo MMS*. É importante lembrar, também, que o valor de $q_3^{P_{st}}$ deve ser sempre o valor ótimo associado ao caminho P_{st}, segundo (P_3).

Em Hansen [19], mostra-se que os problemas MaxMin e MinMax são equivalentes, ou seja, um problema pode ser transformado no outro.

Considerando (P_1) e (P_2) como sendo qualquer um destes dois últimos problemas citados, temos que os problemas da classe de PMC tri-objetivo definida por:

$$(P_1) - (P_2) - (P_3),$$

podem ser resolvidos por variantes do *Algoritmo MMS*.

3.2 PMC tri-objetivo com restrições

Segue-se um PMC tri-objetivo com restrições:

$$\begin{aligned}
(P_r) \quad & \textit{maximizar} && \min_{(i,j)\in AP_{st}} \{p^1(i,j)\} \\
& \textit{minimizar} && \max_{(i,j)\in AP_{st}} \{p^2(i,j)\} \\
& \textit{minimizar} && \sum_{(i,j)\in AP_{st}} p^3(i,j) \\
& \textit{sujeito a:} && \min_{(i,j)\in AP_{st}} \{p^4(i,j)\} \geq \alpha \qquad (3.1) \\
& && \max_{(i,j)\in AP_{st}} \{p^5(i,j)\} \leq \beta \qquad (3.2) \\
& && P_{st} \in S_G^{st},
\end{aligned}$$

onde $\alpha, \beta, p^4(i,j), p^5(i,j) \in \Re, \forall (i,j) \in M$. Note que agora temos cinco pesos associados aos arcos do grafo G.

Conforme Berman, Einav e Handler [5], para atender a restrição (3.2) basta deletar de G todos os arcos (i, j) tal que $p^5(i, j) > \beta$. Veja que todos os caminhos contendo algum arco com peso p^5 maior do que β são inviáveis para (P_τ). Raciocínio análogo vale para (3.1). Então, para resolver o problema (P_τ), basta inserir entre os passos *1* e *2* do *Algoritmo MMS*, o passo:

Delete de G todos os arcos (i, j) *tal que* $p^4(i, j) < \alpha$ *ou* $p^5(i, j) > \beta$.

Observe que, assim como na seção anterior, aqui também podemos definir variantes de PMC tri-objetivo (agora com restrições) que podem ser resolvidos por variantes do *Algoritmo MMS*. Para obter essas variantes devemos juntar as modificações citadas nesta seção e na seção anterior.

3.3 Árvore geradora

Para o problema de árvore geradora com um único objetivo temos, para o objetivo MinSum, o algoritmo de Prim [26], por exemplo. Para o objetivo MinMax temos algoritmos, por exemplo, em Camerini [6] e Gabow e Tarjan [15]. Para o problema de árvore geradora bi-objetivo, com um objetivo MinMax e outro MinSum, temos o algoritmo de Berman, Einav e Handler [5].

Considere o problema MaxMin-MinMax-MinSum apresentado no capítulo anterior. Nesta seção estamos interessados neste problema para árvore geradora. Para este novo problema, algumas adaptações na terminologia do capítulo anterior são necessárias.

Considerando *AlgS* do *Algoritmo MMS* como sendo um algoritmo para o problema de árvore geradora mínima (por exemplo o algoritmo de Prim), e substituindo todos os termos que foram definidos como caminho por árvore geradora do grafo G, tendo como raiz o nó s, temos uma variante do *Algoritmo MMS* para o problema de árvore geradora MaxMin-MinMax-MinSum.

Obviamente que AP_{st} passa a ser o conjunto de arcos da árvore geradora P_{st}, $q_3^{P_{st}}$ o somatório dos pesos (p^3) dos arcos desta árvore e assim por diante.

É fácil ver que a otimalidade desta variante (geração de um conjunto mínimo completo de Pareto-ótimos) é garantida pelo *Teorema 2.3.3* e *Lema 2.3.2*, pois nenhuma modificação adicional é necessária.

Assim como nas duas primeiras seções deste capítulo, aqui também podemos definir problemas de árvore geradora tri-objetivo que pode ser resolvida por variantes do algoritmo apresentado nesta seção.

3.4 Um caso particular do problema (P)

Nesta seção estamos interessados em um caso particular do problema MaxMin-MinMax-MinSum em que é possível reduzir o esforço computacional do *Algoritmo MMS*.

Suponha que no problema (P), apresentado no capítulo anterior, temos que as funções p^1 e p^2 são iguais, ou seja, $p^1(i,j) = p^2(i,j),\ \forall (i,j) \in M$. Evidentemente que o caso em que somente um peso é atribuído a cada arco do grafo recai neste caso particular. Nesta seção vamos apresentar uma variante do *Algoritmo MMS* para este caso particular de (P).

Para este caso particular, $m_1 = m_2 = \hat{m}$ e temos que a complexidade passa para $O\left(\left(\frac{\hat{m}^2}{2} + \frac{\hat{m}}{2}\right) n^2\right)$, ou seja, o número de iterações para problemas grandes reduz-se praticamente à metade. Como $p_1^1 > p_2^1 > \cdots > p_{m_1}^1$ e $p_1^2 < p_2^2 < \cdots < p_{m_2}^2$ (veja (2.1) e (2.2)), então temos:

$$p_1^2 = p_{m_1}^1 < p_2^2 = p_{m_1-1}^1 < \cdots < p_{m_2-1}^2 = p_2^1 < p_{m_2}^2 = p_1^1 \tag{3.3}$$

Como $M_{rg} = \{(i,j) \in M; p^1(i,j) \geq p_r^1 \text{ e } p^2(i,j) \leq p_g^2\}$ e as funções p^1 e p^2 são iguais, então $M_{rg} = \emptyset$ quando $p_r^1 > p_g^2$. Logo não é necessário executar as iterações do *Algoritmo MMS* onde isto acontece.

Observe em (3.3) que para não executar as iterações i e j tais que $p_i^1 > p_j^2$, basta substituir o passo *3.1* pelo passo:

$$Para \;\; j \;\; de \;\; m_2 - i + 1 \;\; até \;\; m_2 \;\; Faça.$$

Com esta única alteração obtemos uma variante do *Algoritmo MMS* para o problema (P) com p^1 e p^2 iguais.

Capítulo 4

Considerações Finais

Este trabalho resultou no desenvolvimento de um algoritmo de complexidade polinomial para um problema de caminho tri-objetivo em Grafos. Para problemas com quato objetivos onde pelo menos três são do tipo gargalo, acreditamos ser possível desenvolver algoritmos polinomiais seguindo a idéia do *Algoritmo MMS*. Porém, antes de enveredar por este caminho achamos ser importante procurar aplicações.

Acreditamos que em algumas aplicações práticas do problema MaxMin-MinMax-MinSum, o número de valores distintos para as funções p^1 e p^2 seja pequeno. A aplicação apresentada no capítulo 2, por exemplo, é um caso onde este fato ocorre. O tempo de execução para problemas com m_1 e m_2 grandes pode ser inviável. Além disso, m_1 e m_2 grandes pode implicar em um número muito grande de soluções Pareto-ótimas (mesmo considerando o conjunto mínimo completo), o que parece não ter sentido prático devido a dificuldade para se tomar a decisão. Por esses motivos, consideramos em nossas implementações poucos valores distintos para p^1 e p^2.

Pudemos observar que a mecânica do *Algoritmo MMS* consiste em inserir arcos ao longo das iterações. Acreditamos que pode ser desenvolvido um algoritmo fazendo o trabalho inverso, ou seja, deletando arcos ao longo das iterações. A vantagem deste novo algoritmo seria, em alguns casos,

executar menos iterações do que o *Algoritmo MMS*. A desvantagem é que as iterações são mais trabalhosas, devido ao teste para identificação de uma nova solução Pareto-ótima. A complexidade (pior caso) destes dois algoritmos é a mesma.

Após a obtenção do conjunto mínimo (ou máximo) completo de Pareto-ótimos, algum procedimento deve ser realizado para a escolha do caminho a ser utilizado. Isto pode ser feito de várias maneiras. Por exemplo, poderia se analizar solução por solução manualmente, ou escolher um Pareto-ótimo atendendo a um outro critério.

Uma importante conclusão dos resultados computacionais é que o número de soluções Pareto-ótimas, tanto do conjunto máximo completo como do conjunto mínimo completo (*cmc*), varia pouco com o tamanho do grafo. Por exemplo, para $Prob_1$ onde $|N| = n$ e aproximadamente $|M| = n^2/10$, o *cmc* aumentou cerca de dez vezes quando n variou vinte vezes, passando de $n = 50$ para $n = 1000$. Para o mesmo problema, *cmc* dobrou quando n variou cinco vezes, passando de $n = 1000$ para $n = 5000$. Resultados similares podem ser verificados com relação à variação de m. Por exemplo, na tabela 2.1 *cmc* variou cerca de quatro vezes quando passamos de $Prob_1$ para $Prob_3$, isto é, aumentamos em aproximadamente dez vezes o número de arcos, passando de $m = n^2/10$ para $m = n^2 - n$. Nas outras tabelas nem isto se verificou, ou seja, foi muito pequena a variação de *cmc* com o aumento do número de arcos. A variação maior de *cmc* parece acontecer com a variação de m_1 e m_2. Estes resultados podem ter importantes conseqüências práticas, significando que enquanto o número de pesos distintos m_1 e m_2 se mantiver dentro de limites razoáveis, o grafo pode crescer sem afetar por demais o número de soluções Pareto-ótimas.

A geração de um número grande de soluções Pareto-ótimas parece não fazer muito sentido na prática, pois leva a tomada de decisão a um

dilema. Pela *Proposição 2.1.2*, *cmc* é limitado por $m_1 m_2$. Além disso, pelo que foi observado no parágrafo anterior sobre os resultados computacionais, o acréscimo de *cmc* parece estar relacionado principalmente com m_1 e m_2. Ou seja, valores pequenos para m_1 e m_2 implicam em *cmc* também pequeno, o que facilita a tomada de decisão. Para valores de m_1 e m_2 pequenos, pudemos observar nas tabelas 2.1, 2.2 e 2.3 que o *Algoritmo MMS* é bastante eficiente com relação ao tempo de execução. Como já foi mencionado, acreditamos que existe um número razoável de aplicações com m_1 e m_2 pequenos.

Referências Bibliográficas

[1] Ahuja, R. K., Magnanti, T. L., Orlin, J. B. "Network flows". New Jersey, 1993.

[2] Azevedo, J.A., Martins, E.Q.V. "An algorithm for the multiobjective shortest path problem on acyclic networks". Investigação Operacional, v. 11 (1), pp. 52–69, 1991.

[3] Batta, R., Chiu, S. S. "Optimal Obnoxious Paths on a Network: Transportation of Hazardous Materials". *Operations Research*, v. 36, pp. 84–92, 1988.

[4] Bellman, R. E. "On a routing problem". *Quart. Appl. Math.*, v. 16, pp. 87–90, 1958.

[5] Berman, O., Einav, D., Handler, G. "The constrained bottleneck problem in network". *Operations Research*, v. 38, pp. 178–181, 1990.

[6] Camerini, P. M. "The min-max spanning tree problem and some extensions". *Information Processing Letters*, v. 7, pp. 10–14, 1978.

[7] Clímaco, J.C.N., Martins, E.Q.V. "On the determination of the nondominated paths in a multiobjective network problem". In: *Methods in Operations Research*, (Anton Hain), v. 40, pp. 255–258, 1981.

[8] Clímaco, J.C.N., Martins, E.Q.V. "A bicriterion shortest path algorithm". *European Journal of Operational Research*, v. 11, pp. 399–404, 1982.

[9] Corley, H.W., Moon, I.D. "Shortest paths in networks with vector weights". *Journal of Optimization Theory and Applications*, v. 46, pp. 79–86, 1985.

[10] Current, J. R., Min, H. "Multiobjectiûe Design of Transportation Networks: Taxonomy and Annotation". *European Journal of Operational Research*, v. 26, 1986.

[11] Current, J. R., Revelle, C.S., Cohon, J.L. "The median shortest path problem: A multiobjective approach to analyze cost vs. accessibility in the design of transportation networks". *Transportation Science*, v. 21 (3), pp. 188–197, 1987.

[12] Current, J. R., Marsh, M. "Multiobjectiûe Transportation Network Design and Routing Problems: Taxonomy and Annotation". *European Journal of Operational Research*, v. 103, pp. 426–438, 1993.

[13] Dijkstra, E. W. "A note on two problems in connexion with graphs". *Numer. Math.*, v. 1, pp. 269–271, 1959.

[14] Ehrgott M., Gandibleux, X. "A survey and annotated bibliography of multiobjective combinatorial optimization". *OR Spektrum*, v. 22, pp. 425–460, 2000.

[15] Gabow, H. N., Tarjan, R. E. "Algorithms for two bottleneck optimization problems". *Journal of Algorithms*, v. 9, pp. 411–417, 1988.

[16] Gandibleux, X., Beugnies, F., Randriamasy, S. "Martins' algorithm revisited for multi-objective shortest path problems with a MaxMin cost function". *A Quarterly Journal of Operations Research*, v. 4, pp. 47–59, 2006.

[17] Guerriero, F., Musmanno R. "Label correcting methods to solve multicriteria shortest path problems". *Journal of Optimization Theory and Applications*, v. 111, n. 3, pp. 589–613, 2001.

[18] Gondran, M., Minoux, M. "Graphs and Algorithms". John Wiley, 1986.

[19] Hansen, P. "Bicriterion path problems". In: *Multicriteria decision making: theory and applications*, Lecture Notes in Economics and Mathematical Systems, v. 177, G. Fandel and T. Gal (eds.), Springer, Heidelberg, pp. 109–127, 1980.

[20] Koopmans, T. C. "Activity Analysis of Production and Allocation". John Wiley & Sons, New York, 1951.

[21] Martins, E. Q. V. "On a multicriteria shortest path problem". *European Journal of Operational Research*, v. 16, pp. 236–245, 1984.

[22] Martins, E. Q. V. "On a special class of bicriterion path problems". *European Journal of Operational Research*, v. 17, pp. 85–94, 1984.

[23] Pareto, V. "Course d'Economic Politique". Lausanne, Rouge, 1896.

[24] Pinto, L. L., Bornstein, C. T. "Algoritmos para problemas de caminho ótimo em grafos". Anais do XXXVIII SBPO (Simpósio Brasileiro de Pesquisa Operacional), Goiânia–GO, pp. 2418–2419, 2006.

[25] Pinto, L. L., Bornstein, C. T. "Software CamOtim". Disponível em: *http://www.cos.ufrj.br/camotim*, 2006.

[26] Prim, R. C. "Shortest connection networks and some generalisations". *Bell System Technical Journal*, v. 36, pp. 1389–1401, 1957.

[27] Tung, C. T., Chew, K. L. "A bicriterion Pareto–optimal path algorithm". *Asia–Pacific Journal of Operational Research*, v. 5, pp. 166–172, 1988.

[28] Tung, C. T., Chew, K. L. "A multicriteria Pareto–optimal path algorithm". *European Journal of Operational Research*, v. 62, pp. 203–209, 1992.

[29] Zeleny, M. "Linear Multiobjective Programming". *Lecture Notes in Economics and Mathematical Systems*, 95, Springer-Verlag, 1974.

[30] Walczak Z., Wojciechowski, J. M. "Transmission Scheduling in Packet Radio Networks using Graph Coloring Algorithm". *International Conference on Wireless and Mobile Communications (ICWMC)*, p. 46, 2006.