ALGORITMO PARA A RESOLUÇÃO DO PROBLEMA DE FLUXO MULTIPRODUTO *FUZZY*

Juliana Verga
Faculdade de Engenharia Elétrica e de Computação - Universidade Estadual de Campinas
Av. Albert Einstein, 400 - Caixa Postal 6101 - CEP:13083-852 - Campinas - SP - Brasil
juverga@dt.fee.unicamp.br

Jussara Rodrigues Ciappina
Faculdade de Engenharia Elétrica e de Computação - Universidade Estadual de Campinas
Av. Albert Einstein, 400 - Caixa Postal 6101 - CEP:13083-852 - Campinas - SP - Brasil
jussara@dt.fee.unicamp.br

Akebo Yamakami
Faculdade de Engenharia Elétrica e de Computação - Universidade Estadual de Campinas
Av. Albert Einstein, 400 - Caixa Postal 6101 - CEP:13083-852 - Campinas - SP - Brasil
akebo@dt.fee.unicamp.br

RESUMO

Este trabalho apresenta uma proposta de solução para o problema de fluxo multiproduto *fuzzy*. O problema foi modelado através de um grafo, cujos nós representam pontos de oferta e demanda de produtos, os quais trafegam pelos arcos da rede. O algoritmo proposto visa encontrar soluções factíveis e boas para o problema de fluxo multiproduto *fuzzy* em redes com incertezas nos custos e capacidades, contendo múltiplas origens e múltiplos destinos.

PALAVRAS-CHAVE. Fluxo Multiproduto *Fuzzy*. Números *Fuzzy*. Programação Matemática *Fuzzy*. Teoria de Grafos.

ABSTRACT

This work presents an approach for solving the fuzzy multicommodity flow problem. The problem was modeled through a graph whose nodes represent points of supply and demand of commodities, which pass through arcs of the network. Our algorithm aims to find a set of good feasible solutions for the fuzzy multicommodity flow problem in networks with uncertainties in the costs and capacities, containing multiple origins and multiple destinations.

KEYWORDS. Fuzzy Multicommodity Flow. Fuzzy Numbers. Fuzzy Mathematical Programming. Graph Theory.

1 Introdução

A teoria dos grafos é comumente utilizada na área da engenharia para resolver problemas que podem ser representados na forma de redes. Dentre diversos problemas abordados, o problema de fluxo multiproduto é um dos que também podem ser modelados por grafos. Este problema surge quando vários produtos compartilham os arcos em uma rede e competem pela capacidade dos mesmos. O objetivo é determinar o fluxo dos produtos em cada arco, a um custo mínimo, de modo a atender as restrições de capacidade dos arcos e as restrições de conservação de fluxo (Chagas, 2005).

As restrições de capacidade dos arcos limitam o fluxo dos produtos, de modo que em nenhum arco trafegue uma quantidade de produtos superior à capacidade suportada por ele. Já as restrições de conservação de fluxo gerenciam o fluxo dos produtos pelos arcos da rede, que saem de um ponto de oferta e chegam em um ponto de demanda. Cada produto pode ser transportado de um ou vários nós origens para um ou vários nós destinos da rede (grafo) (Chagas, 2005).

Os problemas de fluxo multiproduto, de um modo geral, têm recebido muita atenção, devido a aplicabilidade na resolução de problemas atuais presentes nas mais diversas áreas, como transportes, telecomunicações, entre outras. A dificuldade prática de se resolver esse tipo de problema aumenta de forma muito rápida à medida que o mesmo cresce em número de variáveis e, principalmente, em relação ao número de produtos. Aplicações em telecomunicações, por exemplo, tipicamente conduzem a instâncias com um grande número de produtos. Resolver tais instâncias torna-se, portanto, um grande desafio (Chagas, 2005).

Geralmente é associado um custo referente ao trânsito em cada arco, por unidade de fluxo. Estes custos podem ser os mesmos para todos os produtos ou podem ser diferentes para cada produto. Neste trabalho, propomos um algoritmo para abordar problemas de fluxo multiproduto, considerando os mesmos custos nos arcos para todos os produtos.

A proposta de um algoritmo para resolução do problema de fluxo multiproduto *fuzzy* é interessante devido a existência de poucos trabalhos sobre esse assunto e a aplicabilidade desse tipo de algoritmo na resolução de problemas presentes em diversas áreas. A questão da incerteza nos parâmetros de entrada (custos e capacidades), torna o problema mais aderente a realidade, visto que os mesmos são muitas vezes conhecidos aproximadamente, principalmente por causa de informações insuficientes. Por exemplo, os custos, no caso de problemas de transporte, podem depender da distância entre as cidades, das condições da rodovia ou ferrovia (no caso de transporte ferroviário), do tempo gasto para se deslocar entre as cidades, e por isso é considerado um parâmetro incerto. A teoria dos conjuntos *fuzzy* foi utilizada para tratar essas incertezas e oferecer proposta de solução factível e de boa qualidade.

2 Trabalhos correlatos

Os trabalhos iniciais que tratam o problema de fluxo multiproduto foram propostos por Ford e Fulkerson (1962); e Hu (1962), no início dos anos 60. Já para o problema de fluxo multiproduto *fuzzy* são encontrados, na literatura, poucos trabalhos. Dentre eles, podemos citar o de Ghatee e Hashemi (2008) onde os autores propõem dois algoritmos para o problema de fluxo multiproduto *fuzzy*. O primeiro algoritmo utiliza caminhos mínimos *fuzzy* e k-caminhos para gerar os caminhos e encontra o fluxo resolvendo um problema de fluxo de custo mínimo multiproduto clássico. No segundo algoritmo, a oferta e demanda são *fuzzy* e é empregado uma ordem total em números trapezoidais *fuzzy* para reduzir o problema de fluxo multiproduto *fuzzy* em quatro problemas de fluxo multiproduto clássicos.

Hernandes (2007) propôs algoritmos para resolver o problema de caminho mínimo *fuzzy* e também para o problema de fluxo monoproduto de custo mínimo *fuzzy*, que é um caso particular do problema de fluxo multiproduto *fuzzy*. Para o problema de caminhos mínimos *fuzzy*, Hernandes (2007) propôs um algoritmo baseado no algoritmo clássico de Ford-Moore-Bellman (Bellman, 1958). Em tal algoritmo, para a construção do conjunto solução foi utilizado o conceito de dominância de caminhos de Okada e Soper (Okada e Soper, 2000), que será descrita na

próxima seção. Para o problema de fluxo monoproduto de custo mínimo *fuzzy*, Hernandes (2007) propôs alguns algoritmos. O primeiro é uma adaptação do método do Big-M, transformando o problema *fuzzy* em um problema clássico. As demais propostas de solução trabalham com o problema *fuzzy*, onde outros quatro algoritmos foram propostos. O primeiro para o problema de fluxo de custo mínimo *fuzzy* com um nó origem e um nó destino, o segundo para um nó origem e múltiplos nós destinos, o terceiro, para múltiplos nós origens e um nó destino e o último, para múltiplos nós origens e múltiplos nós destinos.

3 Conceitos e definições

Como neste trabalho os parâmetros incertos são tratados utilizando a teoria dos conjuntos *fuzzy*, nesta seção são apresentados alguns conceitos dessa teoria. Para mais detalhes, consultar Dubois e Prade (1980).

No problema estudado, os custos são números triangulares *fuzzy* e as capacidades são números trapezoidais *fuzzy*, somente limitadas superiormente.

Definição 3.1 *Um número triangular fuzzy denotado por* $\tilde{a} = (m, \alpha, \beta)$*, é definido pela seguinte função de pertinência:*

$$\mu_{\tilde{a}}(x) = \begin{cases} 0, & x \leq m - \alpha \\ \frac{x-(m-\alpha)}{\alpha}, & m - \alpha < x < m \\ 1, & x = m \\ \frac{(m+\beta)-x}{\beta}, & m < x < m + \beta \\ 0, & x \geq m + \beta \end{cases} \tag{1}$$

onde m : valor modal, α : espalhamento à esquerda, β : espalhamento à direita.

Os valores $m - \alpha$ e $m + \beta$ são chamados de limitante inferior e superior, respectivamente.

Definição 3.2 *Um número trapezoidal fuzzy denotado por* $\tilde{a} = (m_1, m_2, \alpha, \beta)$ *possui sua função de pertinência* $\mu_{\tilde{a}}(x)$*, definida por:*

$$\mu_{\tilde{a}}(x) = \begin{cases} 0, & x \leq m_1 - \alpha \\ \frac{x-(m_1-\alpha)}{\alpha}, & m_1 - \alpha < x < m_1 \\ 1, & m_1 \leq x \leq m_2 \\ \frac{(m_2+\beta)-x}{\beta}, & m_2 < x < m_2 + \beta \\ 0, & x \geq m_2 + \beta \end{cases} \tag{2}$$

onde m_1 : extremo inferior do valor modal, m_2 : extremo superior do valor modal, α : espalhamento à esquerda e β : espalhamento à direita.

Os valores $m_1 - \alpha$ e $m_2 + \beta$ são chamados de limitante inferior e superior, respectivamente.

Definição 3.3 *Sejam* $\tilde{a}$ *e* $\tilde{b}$ *dois números fuzzy triangulares,* $\tilde{a} = (m_1, \alpha_1, \beta_1)$ *e* $\tilde{b} = (m_2, \alpha_2, \beta_2)$*, a soma desses números é dada por:*

$$\tilde{a} \oplus \tilde{b} = (m_1 + m_2, \alpha_1 + \alpha_2, \beta_1 + \beta_2)\,.$$

A soma de números trapezoidais ocorre de maneira similar.

Uma variedade de métodos para a comparação de números *fuzzy* têm sido propostos na literatura. Alguns deles são: Primeiro índice de Yager (Yager, 1978; 1980; 1981), Índice de Liou e Wang (Liou e Wang, 1992), Índice de García e Lamata (García e Lamata, 2005), Índice de aceitabilidade de Nayeem e Pal (Nayeem e Pal, 2005), dentre outros.

Neste trabalho utilizamos a relação de Okada e Soper, (Okada e Soper, 2000; Okada, 2004) para comparação de números *fuzzy*. Esta relação leva ao conceito de caminho não-dominado ou caminho Pareto-ótimo (Hernandes, 2007).

Definição 3.4 *Sejam $\tilde{a} = (m_1, \alpha_1, \beta_1)$ e $\tilde{b} = (m_2, \alpha_2, \beta_2)$ dois números fuzzy triangulares, então $\tilde{a} \prec \tilde{b}$ ($\tilde{a}$ domina $\tilde{b}$) se e somente se $m_1 \leq m_2$, $(m_1 - \alpha_1) \leq (m_2 - \alpha_2)$, $(m_1 + \beta_1) \leq (m_2 + \beta_2)$ e $\tilde{a} \neq \tilde{b}$.*

Definição 3.5 *Seja $G = (N, A)$ um grafo com custo $\tilde{c} = \{\tilde{c_{ij}}\}$ associado aos seus arcos. Sejam dois subgrafos T^1 e T^2, $T^1 \neq T^2$. O grau de possibilidade de T^2 ser menor que T^1 é dado por (Okada, 2004):*

$$\tilde{w} = Poss\left(\sum_{(i,j)\in T^2} \tilde{c_{ij}} \leq \sum_{(i,j)\in T^1} \tilde{c_{ij}}\right) = \sup \min_x \{\mu_{T^2}(x), \mu_{T^1}(x)\}. \tag{3}$$

onde:

- *Poss: medida de possibilidade;*
- *$\mu_{T^1}(x)$ e $\mu_{T^2}(x)$: são as funções de pertinência dos custos de T_1 e T_2, respectivamente;*
- $\sup \min$*: o valor supremo do mínimo (intersecção), ou seja, o maior grau de pertinência que pode ser obtido do conjunto resultante da intersecção das funções de pertinência $\mu_{T^1}(x)$ e $\mu_{T^2}(x)$.*

Esta equação também foi estudada por Dubois e Prade (1980).

Para encontrar uma solução *fuzzy* utilizando a teoria de possibilidade, é preciso encontrar todas as soluções que possuem algum grau de possibilidade de ser a solução ótima e comparar estas soluções para obter o grau de possibilidade de cada uma (Okada, 2004). O grau de possibilidade de T ser a solução ótima é dado por:

$$D_T = \min_{T^k \in \tau} \left\{ Poss\left(\sum_{(i,j)\in T} \tilde{c_{ij}} \leq \sum_{(i,j)\in T^k} \tilde{c_{ij}}\right)\right\}, \tag{4}$$

onde τ é o conjunto de todas as soluções.

4 Algoritmo proposto

Nesta seção apresentaremos a formulação matemática do problema de fluxo multiproduto de custo mínimo *fuzzy* e o algoritmo proposto.

4.1 Formulação matemática

Seja $G = (N, A)$ um grafo, onde N é o conjunto de nós e A é o conjunto de arcos. Cada arco é denotado por (i, j) onde $i, j \in N$. O problema de fluxo multiproduto de custo mínimo *fuzzy* é formulado como o seguinte problema de programação linear:

$$\begin{aligned} \min \ \ & z = \sum_{k=1}^{K} \sum_{(i,j)\in A} \tilde{c}_{ij}^k x_{ij} \\ \text{s.a} : & \begin{cases} \displaystyle\sum_{j:(i,j)\in A} x_{ij}^k - \sum_{j:(j,i)\in A} x_{ji}^k = b_i^k, \ \forall\, i \in N\,, \ \forall\, k = 1, \ldots, K \\ \tilde{l}_{ij} \leq \displaystyle\sum_{k=1}^{K} x_{ij}^k \leq \tilde{u}_{ij}, \ \forall\, (i,j) \in A \end{cases} \end{aligned} \tag{5}$$

onde:

- x_{ij}^k é o fluxo do produto k que percorre o arco (i, j);
- $\tilde{c}_{ij}^k$ é o custo unitário (o mesmo para todos os produtos) *fuzzy* do produto k associado a cada unidade de fluxo x_{ij} que percorre o arco (i, j);

- $\tilde{l}_{ij}$ é o limitante inferior *fuzzy* da capacidade do arco (i, j);
- $\tilde{u}_{ij}$ é o limitante superior *fuzzy* da capacidade do arco (i, j);
- K é o número total de produtos;
- b_i^k é a oferta ou demanda do produto k:
  - Se $b_i^k > 0$, i é nó gerador do produto k.
  - Se $b_i^k < 0$, i é nó consumidor do produto k.
  - Se $b_i^k = 0$, i é nó de passagem do produto k.

Sem perda de generalidade, os custos são números triangulares *fuzzy*, escritos na forma: (m, α, β) e as capacidades são números trapezoidais *fuzzy* escritos na forma: $(m_1-\alpha, m_1, m_2, m_2+\beta)$, onde $\alpha = m_1 = 0$. Denotamos $(m_1 - \alpha)$ por $\underline{\alpha}$ e $(m_2 + \beta)$ por $\overline{\alpha}$. Logo, a capacidade é escrita na forma: $(\underline{\alpha}, m_1, m_2, \overline{\alpha})$.

4.2 Algoritmo

O algoritmo proposto é baseado no algoritmo de Hernandes (2007), o qual resolve o problema de fluxo monoproduto de custo mínimo *fuzzy*. O algoritmo a seguir, é uma proposta de solução para o problema de fluxo multiproduto de custo mínimo *fuzzy*, com incertezas nos custos e nas capacidades dos arcos.

- Passo 1: Encontrar os caminhos mínimos não-dominados para cada par de nós origem-destino de cada produto;

  Para encontrar todos os caminhos não-dominados para cada par de nós origem-destino de cada produto, utilizamos o algoritmo proposto por Hernandes (2007). Tal algoritmo é baseado no algoritmo clássico de Ford-Moore-Bellman, (Bellman, 1958). Trata-se de um algoritmo iterativo, tendo como critério de parada o número de iterações ou a não alteração dos custos de todos os caminhos encontrados na iteração anterior com relação à iteração atual. Desta forma, são encontrados todos os caminhos não-dominados entre os nós origem e destino de cada produto aplicando a relação de Okada e Soper (2000), descrita pela definição 3.4, para descartar os caminhos dominados. Note que é necessário identificar os caminhos não-dominados encontrados para cada produto, de forma que, ao ordenarmos e enviarmos fluxo, sabemos qual é o produto em questão.

- Passo 2: Ordenar todos os caminhos p_k:

  μ_{cam}=$Poss\{p_k$ ser o melhor caminho$\}$= $\min\{\mu_{custo}, \mu_{capac}\}$, onde:

  1. $\mu_{custo} = Poss\{p_k$ ser mínimo$\}$;
  2. $\mu_{capac} = \min\limits_{(i,j)\in p_k} \{\mu_{capac}(i, j)\} = Poss\{$fluxo passar no arco (i, j) através de $p_k\}$.

  Para fazer a ordenação dos caminhos:

  - Calcular μ_{custo} para cada caminho, ou seja, verificar através da medida de possibilidade de Dubois e Prade (1980), qual a possibilidade de cada caminho ter o custo menor do que todos os demais.
  - Calcular μ_{capac} para cada caminho.
  - Calcular a pertinência de cada caminho: $\mu_{cam} = \min\{\mu_{custo}, \mu_{capac}\}$.
  - Colocar os caminhos em ordem decrescente de acordo com μ_{cam}. Se houver empate, usar o valor modal dos custos dos caminhos: escolher aquele com menor custo. Se ainda empatar, assumir a ordem que aparece.

- Passo 3: Envio de fluxo:

1. Enviar fluxo para o primeiro caminho ordenado até alcançar a próxima pertinência diferente da pertinência desse caminho.
2. Se o fluxo necessário ao nó destino do caminho em questão já foi satisfeito, eliminar todos os caminhos do produto em questão que tenham tal nó como nó destino.
3. Atualizar μ_{capac}, $\forall\, p_k$.

Para enviar fluxo pelos caminhos de acordo com a ordenação feita:

- Verificar se existe oferta no nó origem do melhor caminho, caso não haja, ir para o próximo melhor caminho com oferta.
- Encontrar a próxima pertinência diferente (se houver) a do melhor caminho (denotaremos por μ_{relax}) para calcular o fluxo a ser enviado nos arcos, caso esta seja zero e a pertinência do melhor caminho for diferente de zero, usamos ela própria.

Observação 1 *Conforme as demandas forem atendidas eliminamos os caminhos correspondentes, podendo ocasionar a sobra de somente um caminho, neste caso, relaxamos as pertinências dos arcos até o limitante superior. Fazemos o mesmo quando temos vários caminhos e somente no último há oferta no nó origem.*

- Para calcular o fluxo a ser enviado, primeiramente calculamos o fluxo permitido em cada arco do caminho em questão, usando μ_{relax}. A seguir, calculamos a diferença entre o fluxo permitido e o fluxo já existente para cada arco do caminho e tomamos o mínimo.
- Se $\mu_{relax} = \mu_{cam}$, então pode acontecer de $\mu_{cam} = \mu_{capac}$ e portanto o fluxo permitido será igual ao fluxo presente no arco, logo não enviamos fluxo algum. Neste caso, utilizamos o próximo melhor caminho para enviar fluxo ou a seguinte opção:

  Opção 1. Recalcular o fluxo da seguinte forma:

  $$\min_{(i,j)\in p_k} \left\{ \frac{\overline{\alpha} \text{ do arco}(i,j) - \text{fluxo no arco}(i,j)}{2} \right\}.$$

  Assim, enviaremos uma certa quantidade de fluxo, a qual garante que não alcançará o limitante superior de nenhum arco do caminho p_k.
- Eliminar os caminhos cuja demanda foi atendida.
- Atualizar μ_{capac}.

- Passo 4: Critério de parada.
  1. Se existir fluxo a transitar ir para o Passo 2.
  2. Senão $\Rightarrow$ fim.

5 Testes Computacionais

Nesta seção, dois exemplos são apresentados para ilustrar o algoritmo proposto, implementado em Matlab 7.0.1 e executado em uma plataforma Intel Core Duo, 1.73 GHz e 2 Gb de RAM.

O primeiro exemplo é apresentado na seção 5.1. É uma rede pequena contendo seis nós e nove arcos, com finalidade didática pois permite uma análise detalhada.

O segundo exemplo é apresentado na seção 5.2, maior que o primeiro, com finalidade de mostrar a funcionalidade e eficiência do algoritmo proposto.

A análise dos resultados será feita baseando-se no fluxo final de cada produto, custo final de cada produto e na pertinência final. O custo final é calculado através da multiplicação dos custos dos arcos com a quantidade de fluxo percorrido em cada arco. A pertinência final é o valor mínimo dentre as pertinências dos arcos.

Observação 2 *Os exemplos aqui apresentados são pequenos, mas para exemplos maiores o comportamento do algoritmo apresentado é semelhante.*

5.1 Exemplo 1

O primeiro exemplo foi resolvido utilizando o grafo apresentado na Figura 1, representado pelos dados da Tabela 1. Consideramos dois produtos, o produto 1 (p_1) tem como origem o nó 1 e destinos os nós 4, 5 e 6 e o produto 2 (p_2) tem como nó origem o nó 2 e destinos os nós 4, 5 e 6. As ofertas e as demandas dos produtos são: $b_1^1 = 5$, $b_2^2 = 2$, $b_4^1 = -2.3333$, $b_4^2 = -0.6667$, $b_5^1 = -1.3333$, $b_5^2 = -0.6667$, $b_6^1 = -1.3333$ e $b_6^2 = -0.6667$.

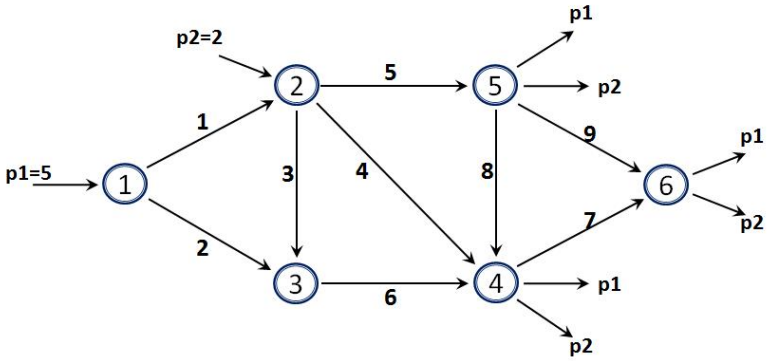

Figura 1: Rede com dois produtos

Tabela 1: Custos e capacidades da rede da Figura 1

Arcos	Custos	Capacidades
1	(2, 1, 1)	(0, 0, 3, 4)
2	(3, 2, 1)	(0, 0, 1.5, 2.5)
3	(3, 3, 2)	(0, 0, 2, 3)
4	(6, 4, 2)	(0, 0, 2, 4)
5	(5, 1, 1)	(0, 0, 7, 8)
6	(5, 2, 1)	(0, 0, 3, 4)
7	(2, 1, 1)	(0, 0, 4, 6)
8	(5, 4, 2)	(0, 0, 4, 5)
9	(3, 1, 1)	(0, 0, 2, 3)

O algoritmo atendeu todas as demandas em nove etapas utilizando seis caminhos, dos dez caminhos mínimos não-dominados. O tempo de processamento foi de 1.5 segundos.

O envio de fluxo ocorreu segundo a Tabela 2.

Tabela 2: Envio de fluxo dos produtos

etapa	caminho	produto	fluxo enviado	demanda atendida
1	$2 \rightarrow 5$	p_2	0.6667	0.6667
2	$2 \rightarrow 4$	p_2	0.6667	0.6667
3	$1 \rightarrow 2 \rightarrow 4$	p_1	2.3333	2.3333
4	$2 \rightarrow 4 \rightarrow 6$	p_2	0.3333	0.3333
5	$1 \rightarrow 2 \rightarrow 5$	p_1	1.3333	1.3333
6	$2 \rightarrow 4 \rightarrow 6$	p_2	0.0952	0.4285
7	$2 \rightarrow 4 \rightarrow 6$	p_2	0.2381	0.6667
8	$1 \rightarrow 2 \rightarrow 4 \rightarrow 6$	p_1	0	0
9	$1 \rightarrow 3 \rightarrow 4 \rightarrow 6$	p_1	1.3333	1.3333

Podemos observar na Tabela 2 que não foi enviado fluxo do produto p_1 na etapa oito. Neste caso, podemos usar a Opção 1, citada anteriormente, enviando fluxo pelo melhor caminho segundo a ordenação feita. Desse modo, as etapas oito e nove mudaram segundo a Tabela 3.

Tabela 3: Envio de fluxo com a Opção 1

etapa	caminho	produto	fluxo enviado	demanda atendida
8	$1 \rightarrow 2 \rightarrow 4 \rightarrow 6$	p_1	0.1667	0.1667
9	$1 \rightarrow 3 \rightarrow 4 \rightarrow 6$	p_1	1.1667	1.3333

O fluxo final dos produtos p_1 e p_2 em cada arco, segue nas Tabelas 4 e 5.

Tabela 4: Fluxo final de cada produto sem a Opção 1

arcos	(1,2)	(1,3)	(2,3)	(2,4)	(2,5)	(3,4)	(4,6)	(5,4)	(5,6)
fluxo de p_1	3.6667	1.3333	0	2.3333	1.3333	1.3333	1.3333	0	0
fluxo de p_2	0	0	0	1.3333	0.6667	0	0.6667	0	0
fluxo total	3.6667	1.3333	0	3.6667	2.0000	1.3333	2.0000	0	0

Tabela 5: Fluxo final de cada produto com a Opção 1

arcos	(1,2)	(1,3)	(2,3)	(2,4)	(2,5)	(3,4)	(4,6)	(5,4)	(5,6)
fluxo de p_1	3.8333	1.1667	0	2.5000	1.3333	1.1667	1.3333	0	0
fluxo de p_2	0	0	0	1.3333	0.6667	0	0.6667	0	0
fluxo total	3.8333	1.1667	0	3.8333	2.0000	1.1667	2.0000	0	0

Podemos observar na tabela 4, que no arco (1,2) tem 3.6667 de fluxo e no arco (1,3) tem 1.3333. Isso se deve ao fato do arco (1,2) pertencer aos melhores caminhos segundo a ordenação feita. Para p_1, os arcos (1,2) e (2,4) foram utilizados para atender toda a demanda do nó 4 (enviando 2.3333 pelo caminho $1 \rightarrow 2 \rightarrow 4$) com custo (8 5 3), de outra forma custaria (10 6 4) correspondente ao caminho $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$, (12 6 4) correspondente ao caminho $1 \rightarrow 2 \rightarrow 5 \rightarrow 4$ ou (8 4 2) correspondente ao caminho $1 \rightarrow 3 \rightarrow 4$. Para p_2, o arco (2,4) foi utilizado para atender toda demanda do nó 4 (enviando 0.6667 por $2 \rightarrow 4$) com custo (6 4 2), enquanto que as outras opções custariam (8 5 3) correspondente ao caminho $2 \rightarrow 3 \rightarrow 4$ ou (10 5 3) correspondente ao caminho $2 \rightarrow 5 \rightarrow 4$. Portanto, os caminhos utilizados foram boas opções.

Neste exemplo, a opção de usar o melhor caminho (Opção 1), não trouxe vantagens. O custo final sem a opção foi (54.0000, 27.6667, 17.6667) e a pertinência 0.1667, referente ao arco $(2, 4)$; enquanto que, com a opção o custo foi (54.0000, 27.8333, 17.8333) e a pertinência 0.0833 devido ao fluxo no arco $(2, 4)$ ter aumentado. A tabela 6 ilustra o custo final de cada produto sem a Opção 1 .

Tabela 6: Custo final de cada produto

custo final	produto
(41.3333, 21.0000, 13.6667)	p_1
(12.6667, 6.6667, 4.0000)	p_2

Para resolver o problema clássico, ou seja, quando os custos e capacidades são números reais, utilizamos o solver do Xpress, que é um pacote de resolução de programação linear que resolve através do simplex. Os custos são os valores modais dos custos da Tabela 1 e as capacidades, o limitante superior das capacidades da Tabela 1. Os resultados seguem na Tabela 7.

Tabela 7: Fluxo final de cada produto

arcos	(1,2)	(1,3)	(2,3)	(2,4)	(2,5)	(3,4)	(4,6)	(5,4)	(5,6)
fluxo de p_1	2.6667	2.3333	0	0	2.6667	2.3333	0	0	1.3333
fluxo de p_2	0	0	0	1.3333	0.6667	0	0.6667	0	0
fluxo total	2.6667	2.3333	0	1.3333	3.3334	2.3333	0.6667	0	1.3333

Os custos finais de p_1 e p_2 foram respectivamente, 41.3332 e 12.6667. O custo final total foi 53.9999.

Analisando as soluções do problema clássico e do problema *fuzzy*, observamos que as mesmas são próximas.

5.2 Exemplo 2

O segundo exemplo foi resolvido utilizando a Rede Óptica Européia COST239 (Tan e Sinclair, 1995) apresentada pela Figura 2. As capacidades foram adaptadas para números *fuzzy*. Os custos, a definição dos nós e arcos foram retirados de Hernandes (2007).

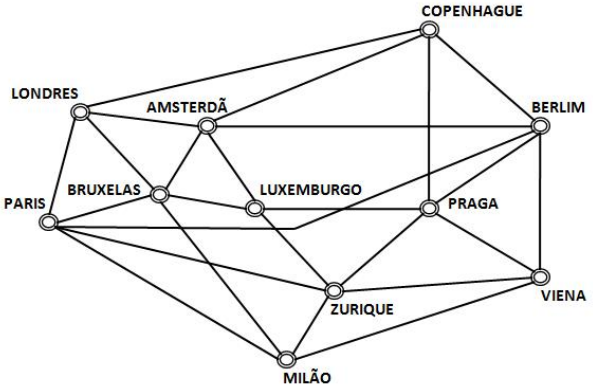

Figura 2: Rede Óptica Européia

Os arcos, custos e capacidades estão definidos na Tabela 8. Neste problema, consideramos os arcos com duplo sentido sendo que, os arcos de 26 a 50 correspondem aos arcos de 1 a 25 em sentido contrário.

Tabela 8: Dados da Rede COST239

Arco	Origem → Destino	Custos	Capacidades
1	1 → 2	(820, 20, 20)	(0, 0, 10, 12)
2	1 → 3	(361, 11, 9)	(0, 0, 2.5, 4.5)
3	1 → 6	(677, 27, 6)	(0, 0, 10, 12)
4	1 → 9	(300, 10, 50)	(0, 0, 4.0, 6.5)
5	1 → 10	(450, 30, 20)	(0, 0, 2.5, 3.5)
6	2 → 3	(186, 6, 7)	(0, 0, 30, 34)
7	2 → 5	(510, 15, 15)	(0, 0, 20, 23)
8	2 → 9	(930, 30, 30)	(0, 0, 30, 34)
9	3 → 4	(667, 17, 196)	(0, 0, 10, 12)
10	3 → 5	(748, 18, 22)	(0, 0, 10, 12)
11	3 → 8	(443, 18, 22)	(0, 0, 2.5, 3.5)
12	4 → 5	(199, 9, 11)	(0, 0, 20, 23)
13	4 → 6	(340, 30, 20)	(0, 0, 10, 12)
14	4 → 11	(740, 30, 30)	(0, 0, 10, 12)
15	5 → 6	(660, 50, 30)	(0, 0, 30, 34)
16	6 → 11	(242, 12, 18)	(0, 0, 10, 12)
17	7 → 6	(410, 20, 30)	(0, 0, 20, 23)
18	7 → 11	(472, 22, 18)	(0, 0, 10, 12)
19	8 → 4	(730, 20, 5)	(0, 0, 2.5, 3.5)
20	8 → 7	(242, 12, 13)	(0, 0, 2.5, 3.5)
21	9 → 8	(137, 7, 8)	(0, 0, 2.5, 3.5)
22	9 → 7	(130, 10, 20)	(0, 0, 10, 12)
23	9 → 10	(242, 12, 18)	(0, 0, 2.5, 3.5)
24	10 → 7	(342, 12, 8)	(0, 0, 2.5, 3.5)
25	10 → 11	(1310, 60, 120)	(0, 0, 2.5, 3.5)

Os nós da rede estão definidos na Tabela 9.

Tabela 9: Denominação dos nós da rede COST239

Nós	Cidades	Nós	Cidades
1	Paris	7	Amsterdã
2	Milão	8	Luxemburgo
3	Zurique	9	Bruxelas
4	Praga	10	Londres
5	Viena	11	Copenhague
6	Berlim		

Consideramos três produtos, os quais estão descritos na Tabela 10 (em $Gbits/s$).

Tabela 10: Dados de cada produto

produto	origem	destinos	oferta	demanda
p_1	1	{3, 6, 11}	6	{0.5, 2.5, 3}
p_2	1	{2, 5, 10}	5	{2, 0.5, 2.5}
p_3	1	{4, 7, 9}	7	{4, 1, 2}

O algoritmo atendeu todas as demandas em nove etapas utilizando oito caminhos, dos dez caminhos mínimos não-dominados. O tempo de processamento foi de 7 segundos.
O envio de fluxo ocorreu segundo a Tabela 11.

Tabela 11: Envio de fluxo

etapa	caminho	produto	fluxo enviado	demanda atendida
1	$1 \rightarrow 9$	p_3	2.0	2.0
2	$1 \rightarrow 3$	p_1	0.5	0.5
3	$1 \rightarrow 9 \rightarrow 7$	p_3	1.0	1.0
4	$1 \rightarrow 10$	p_2	2.5	2.5
5	$1 \rightarrow 3 \rightarrow 2$	p_2	2.0	2.0
6	$1 \rightarrow 6$	p_1	2.5	2.5
7	$1 \rightarrow 9 \rightarrow 7 \rightarrow 11$	p_1	3.0	3.0
8	$1 \rightarrow 6 \rightarrow 4$	p_3	4.0	4.0
9	$1 \rightarrow 3 \rightarrow 2 \rightarrow 5$	p_2	0.5	0.5

Podemos observar na Tabela 11 que para cada par de nós origem-destino de cada produto, as demandas foram atendidas usando apenas um caminho. As Tabelas 12, 13 e 14, ilustram o fluxo final dos produtos p_1, p_2 e p_3, respectivamente e, a tabela 15, o fluxo total nos arcos.

Tabela 12: Fluxo final do produto p_1

arcos	2	3	4	18	22
fluxo de p_1	0.5000	2.5000	3.0000	3.0000	3.0000

Tabela 13: Fluxo final do produto p_2

arcos	2	5	7	31
fluxo de p_2	2.5000	2.5000	0.5000	2.5000

Tabela 14: Fluxo final do produto p_3

arcos	3	4	22	38
fluxo de p_3	4.0000	3.0000	1.0000	4.0000

Tabela 15: Fluxo total nos arcos

arcos	2	3	4	5	7	18	22	31	38
fluxo total	3.0000	6.5000	6.0000	2.50000	0.5000	3.0000	4.0000	2.5000	4.0000

O custo final total foi $10^4 \cdot (1.2425,\ 0.0592,\ 0.0655)$ e a pertinência 0.2000, referente ao arco $(1, 9)$.

A Tabela 16 ilustra o custo final de cada produto.

Tabela 16: Custo final de cada produto

custo final	produto
$10^3 \cdot (4.5790,\ 0.1919,\ 0.2835)$	p_1
$10^3 \cdot (2.7475,\ 0.1250,\ 0.0975)$	p_2
$10^3 \cdot (5.0980,\ 0.2680,\ 0.2740)$	p_3

Para o problema clássico, ou seja, quando os custos e capacidades são números reais, utilizando o solver do Xpress, obtivemos os resultados apresentados nas Tabelas 17, 18 e 19. Os custos são os valores modais dos custos da Tabela 8 e as capacidades, os valores modais das capacidades da Tabela 8.

Tabela 17: Fluxo final do produto p_1

arcos	2	3	4	18	22
fluxo de p_1	0.5000	2.5000	3.0000	3.0000	3.0000

Tabela 18: Fluxo final do produto p_2

arcos	2	5	7	31
fluxo de p_2	2.5000	2.5000	0.5000	2.5000

Tabela 19: Fluxo final do produto p_3

arcos	3	4	22	38
fluxo de p_3	4.0000	3.0000	1.0000	4.0000

Os custos finais de p_1, p_2 e p_3 foram respectivamente, 4579, 2747.5 e 5098. O custo final total foi 12.425. Os resultados foram iguais aos do problema *fuzzy*, porque as capacidades foram suficientes, ou seja, as restrições de capacidade não foram utilizadas.

6 Conclusões e trabalhos futuros

Problemas de redes têm sido extensamente estudados e aplicados nas soluções de problemas reais. Neste trabalho apresentamos um algoritmo para a resolução do problema de fluxo multiproduto *fuzzy* que aborda incertezas em dois parâmetros (custos e capacidades) e possui a

vantagem de ser aplicável a problemas de grande porte. A análise dos resultados dos testes computacionais mostra que o algoritmo proposto apresentou resultados satisfatórios. A comparação com o problema resolvido na forma clássica, mostra que os resultados, de fato, foram bons.

O uso da teoria *fuzzy* para tratar as incertezas nos custos e nas capacidades tornou o algoritmo mais aderente à realidade e promissor para a resolução do problema apresentado, pois resolvemos o problema na forma *fuzzy*, que é mais interessante do que transforma-lo na forma clássica para resolve-lo.

Nos dois exemplos apresentados, temos valores de demanda que não são inteiros: no primeiro exemplo, consideramos que as unidades dos produtos podem ser particionadas para destinos diferentes. Já no segundo exemplo, a medida de tráfego foi dada em termos de Gbit/s (Tan e Sinclair, 1995). Um trabalho futuro possível é tratar a integralidade, ou seja, trabalhar apenas com valores inteiros nas ofertas e demandas dos produtos. Outro aspecto que merece ser abordado é considerar custos distintos nos arcos para cada produto. Além disto, para problemas de grande porte, pretendemos estudar métodos utilizando técnicas de decomposição.

Agradecimentos

Este trabalho foi financiado pelas agências CAPES e CNPq.

Referências

Bellman, R.E. (1958), On a routing problem, *Quarterly Applied Mathematics*, 16, 87-90.

Chagas, R.J., *Uma Aplicação de Métodos Aproximados a Problemas de Fluxo Multiproduto Inteiro*, Dissertação de Mestrado, CEFET-MG, outubro, 2005.

Dubois, D. & Prade, H., *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, INC, New York, 1980.

Ford, L.R. & Fulkerson, D.R., *Flows in networks*, Princeton University Press, New York, 1962.

García, M.S. & Lamata, M.T. (2005), The fuzzy sets in maintenance process, *Proceedings of the European Society for Fuzzy Logic and Technology*, 118-123.

Ghatee, M. & Hashemi, S.M. (2008), Some concepts of the fuzzy multicommodity flow problem and their application in fuzzy network design, *Mathematical and Computer Modelling*, 33, 344-360.

Hernandes, F., *Algoritmos para Problemas de Grafos com Incertezas*, Tese de doutorado, Faculdade de Engenharia Elétrica e Computação, UNICAMP, Fevereiro, 2007.

Hu, T.C. (1962), Multicommodity network flows, *Operations Research*, 11, 344-360.

Liou, T.S. & Wang, M.J. (1992), Ranking fuzzy numbers with integral interval, *Fuzzy Sets and Systems*, 50, 247-255.

Nayeem, S.M.A. & Paul, M. (2005), Shortest path problem on a network with imprecise edge weight, *Fuzzy Optimization and Decision Making*, 4, 293-312.

Okada, S. & Soper, T. (2000), A shortest path problem on a network with fuzzy arc lengths, *Fuzzy Sets and Systems*, 109, 129-140.

Okada, S. (2004), Fuzzy shortest path problems incorporating interactivity among paths, *Fuzzy Sets and Systems*, 142(3), 335-357.

Tan, L.G. & Sinclair M.C., Wavelength assignment between the central nodes of the cost239 european optical network, *11th UK Performance Engineering Workshop*, Liverpool, pp. 235-247, 1995.

Yager, R.R. (1978), Ranking fuzzy subsets over the unit interval, *Proceedings of the CDC*, 1435-1437.

Yager, R.R. (1980), On choosing between fuzzy subsets, *Kybernects*, 9, 151-154.

Yager, R.R. (1981), A procedure for ordering fuzzy subsets of the unit interval, *Informations Sciences*, 24, 143-161.