Universidade de Coimbra
Faculdade de Economia

# Problemas de Fluxos em Redes, com Objectivos Múltiplos

Augusto Manuel José Eusébio

Tese apresentada na Faculdade de Economia da Universidade de Coimbra para a obtenção do grau de Doutor, na área de Organização e Gestão de Empresas, especialidade de Investigação Operacional, sob orientação do Professor Doutor José Rui Figueira.

Coimbra
2009

À memória da minha mãe
À Olga, à Inês e ao Pedro

# Agradecimentos

Ao Professor Doutor José Rui Figueira pelo tempo que generosamente me dedicou transmitindo-me os melhores e mais úteis ensinamentos, com paciência e confiança. Pelo acesso que me facilitou a uma pesquisa mais alargada e enriquecedora e pela sua crítica sempre tão atempada e construtiva. Pela motivação e pelo incentivo que sempre me transmitiu que me levaram a percorrer novos caminhos. Pela sua amizade e compreensão nos momentos mais difíceis. Bem-haja estou-lhe muito, muito grato.

Ao Professor Doutor João Lisboa, ao Professor Doutor Helder Coelho e ao saudoso Professor Doutor Pedro Matos pelas suas palavras de motivação inicial e pela confiança que depositaram em mim.

À Dr.a Rosinda Pimentel pela leitura e sugestões de correcção no texto desta Tese.

Ao Centro de Estudos de Gestão do Instituto Superior Técnico (CEG-IST) por me ter integrado como investigador.

Ao Instituto Politécnico de Leiria em particular ao seu presidente Professor Doutor Luciano Almeida pela dispensa de serviço que me foi concedida.

À Escola Superior de Tecnologia e Gestão em particular aos membros do Conselho Científico pela confiança depositada, sem a qual não teria sido possível a dispensa de serviço.

À Biblioteca do Departamento de Matemática da Universidade de Coimbra pela disponibilização de uma grande parte da bibliografia utilizada na preparação desta Tese.

À minha família, por todo o apoio, carinho e amor, especialmente à minha esposa, por suportar pacientemente o trabalho extra, resultante da minha ausência.

A todos agradeço, profundamente, e dedico o resultado deste trabalho.

# Resumo

O trabalho apresentado nesta Tese centra-se no domínio da optimização multi-objectivo, mais concretamente dos problemas de fluxos em redes com dois ou mais objectivos, desenvolvendo-se um conjunto de algoritmos originais, para a determinação de soluções eficientes e não-dominadas em problemas de fluxos em redes. Começámos por fazer uma revisão da bibliografia na área dos problemas de fluxos em redes, onde descrevemos os algoritmos exactos e os algoritmos aproximados, para problemas de fluxos em redes multi-objectivo com variáveis contínuas e com variáveis inteiras. Nesta revisão apercebemo-nos de que a maioria dos trabalhos existentes para este tipo de problemas se debruça apenas sobre o caso de problemas com dois objectivos. Além disso, percebemos também que existem nestes casos várias dificuldades. Partimos em busca de uma melhor compreensão deste tipo de problemas. Descobrimos também que um dos principais métodos utilizados na bibliografia, para calcular as soluções não-dominadas suportadas, afinal não calculava todas as soluções suportadas. Apresentámos exemplos que provam este facto. Propusemos um conjunto de novos algoritmos: um algoritmo do tipo primal-dual para o cálculo das soluções não-dominadas extremas no caso do problema de fluxos em redes bi-objectivo; um algoritmo baseado nos ciclos de custo zero, para o cálculo de todas as soluções eficientes ou não-dominadas suportadas do problema de fluxos em redes com variáveis inteiras multi-objectivo (este algoritmo resultou da demonstração que fizemos da conexidade deste conjunto de soluções, apresentada nesta Tese); uma versão melhorada do algoritmo de restrição-$\varepsilon$ que calcula todas as soluções eficientes ou não-dominadas do problema de fluxos em redes com variáveis inteiras multi-objectivo; e um algoritmo dos trapézios, para o cálculo de representações do conjunto de todas as soluções não-dominadas, no problema de fluxos em redes inteiro bi-objectivo. Todos os algoritmos apresentados foram implementados em linguagem de programação `C` e os resultados foram apresentados e analisados.

# Abstract

The work presented in this Thesis is devoted to the field of multi-objective network flows problems. It begins with a survey of all known exact and approximate algorithms for continuous network flow problems as well as integer network flow problems. We began by doing a survey of all the algorithms known for solving the multiple objective flow problems, for both the continuous and integer case exact and approximation algorithms. We observed that the large majority of the algorithms were designed for taking into account only two objectives and there were several algorithms incorrect. We presented some examples showing that the most used method to find all the supported non-dominated solutions for the integer bi--objective network flow problem was wrong. A set of original algorithms were proposed: a primal-dual algorithm that finds all the extreme non-dominated solutions for the bi-objective network flow problem, a primal-dual algorithm for the minimum flow problem, a cost zero cycle algorithm that finds all the supported efficient/non-dominated solutions for the integer multi-objective network flow problem (this algorithm is based on the proof of the connectedness of the supported non-dominated solutions that was presented also in this Thesis), an improved $\varepsilon$-constraint algorithm that finds all efficient/non-dominated solutions for the integer bi-objective network flow problem and a trapezium algorithm that finds representations of the set of all non-dominated solutions for the integer bi-objective network flow problem. All the algorithms have been implemented by using the `C` programming language and the results and analysis were reported in this document too.

# Lista de siglas e notação

## Siglas

| | |
|---|---|
| EAG | Estrutura de árvore geradora. |
| EF(X) | Conjunto de soluções eficientes, para o problema com região admissível X. |
| EFE(X) | Conjunto de soluções eficientes extremas, para o problema com região admissível X. |
| ES | Eficiente Suportada. |
| FCM | Fluxo de custo mínimo. |
| FICM | Fluxo inteiro de custo mínimo. |
| FRBO | Fluxos em redes bi-objectivo. |
| FRIBO | Fluxos em redes inteiro bi-objectivo. |
| FRIMO | Fluxos em redes inteiro multi-objectivo. |
| FRMO | Fluxos em redes multi-objectivo. |
| FRITO | Fluxos em redes inteiro tri-objectivo. |
| ND | Não-dominada. |
| NDS | Não-dominada suportada. |
| ND(Y) | Conjunto de soluções não-dominadas extremas, para o problema com região admissível Y, no espaço dos objectivos. |
| PL | Programa Linear. |
| PSAP | Procedimento de separação e avaliação progressivas. |

## Notação

| Notação | Explicação. |
|---|---|
| $a \leqslant b$ | o número real $a$ é menor ou igual que número real $b$. |
| $a \geqslant b$ | o número real é $a$ maior ou igual que número real $b$. |
| $u \leq v$ | as componentes do vector $u$ são menores ou iguais às componentes do vector $v$, respectivas, com pelo menos um par de componentes diferente. |
| $u \leqq v$ | as componentes do vector $u$ são menores ou iguais às componentes do vector $v$, respectivas. |
| $u \geqq v$ | as componentes do vector $u$ são maiores ou iguais às componentes do vector $v$, respectivas. |

# Conteúdo

# Lista de Figuras

# Lista de Tabelas

# Capítulo 1

# Introdução

Os problemas de fluxos em redes têm tido um interesse crescente nas últimas décadas, tanto do ponto de vista teórico como prático Ahuja *et al.* (1993), Lim & Lee (2008). Imaginar um mundo sem problemas de fluxos em redes não é fácil. A sociedade está organizada em redes imprescindíveis em diferentes áreas, por exemplo, na deslocação das pessoas (redes de estradas, de caminhos de ferro, de ligações aéreas, de vias marítimas); para permitir a comunicação entre elas (redes de transporte de correio, redes de telecomunicações, redes de computadores, redes de dados); na construção de um ambiente saudável (redes de recolha de lixo, de distribuição de água, de organização da floresta); no divertimento e lazer (redes de hotéis, cafés, espaços de diversão); na preservação da saúde e tratamento de doenças (redes de hospitais, de meios de socorro, de centros de saúde, de centros de desporto); na formação (redes de escolas, de centros de formação, de espaços de teatro e cinema); e na satisfação das necessidades básicas (redes de produção e distribuição de alimentos, de distribuição de energia).

Em todas estas redes queremos deslocar ou transportar alguma coisa (pessoas, produtos de consumo, informação, energia, objectos, etc.) de determinados pontos para outros de uma forma eficiente, o que origina problemas que são, quase sempre, difíceis de resolver sem a ajuda de ferramentas adequadas, nomeadamente de optimização.

Entre as inúmeras situações, onde a aplicação deste tipo de problemas tem sido feita, destacamos o transporte de pessoas, objectos ou informação; a concessão de habitação em função da alteração das necessidades das pessoas; o corte de placas ou vigas a disponibilizar para futura utilização, de forma a satisfazer as necessidades com o menor desperdício possível; o planeamento de torneios desportivos envolvendo várias equipas; a construção de redes eléctricas; a determinação da melhor política de energia; a procura da melhor distribuição de alunos por escolas minimizando a

distância a percorrer pelos alunos e, ao mesmo tempo, permitindo um equilíbrio racial; a determinação do contorno óptimo dos poços a abrir em minas; a localização de terminais de uma companhia de transporte a instalar entre um conjunto de escolhas possíveis de forma a satisfazer a procura; a destruição de alvos militares; a distribuição de material de reparação de equipamento por diversos locais por forma a que, quando uma reparação for necessária, as peças existam nos locais apropriados; a comparação de sequências de ADN; a determinação da menor duração possível para um projecto como, por exemplo, a construção de um edifício; a construção de planos de evacuação em grandes edifícios; a organização de espaços em armazéns; o planeamento da produção numa empresa; a determinação da matriz de intensidades de radiação a enviar para o tecido cancerígeno de um corpo, minimizando a área de tecido saudável apanhada pela radiação, evitando a perda deste e que o funcionamento de órgãos vitais possa ser afectado.

Estes problemas são todos mono-objectivo, isto é, apenas é considerada a optimização de um objectivo. No entanto, a realidade é por natureza multidimensional e os problemas surgem naturalmente com vários objectivos, que não podem ser convertidos ou amalgamados num único, por diversas razões, entre as quais as diferentes unidades de medida dos objectivos. Não tem sentido o amalgamento de dimensões heterogéneas (distância, barulho, humidade, etc.) numa única unidade, cujo significado é difícil ou impossível de se definir.

Podemos enumerar alguns problemas de fluxos em redes multi-objectivo: (1) num problema de transportes, uma fábrica pretende entregar os seus produtos, em armazéns ou locais de consumo, minimizando o custo de transporte, o tempo de entrega, o consumo de combustível e maximizando a utilização de um determinado processo de fabrico (ver Steuer, 1986); (2) os gerentes de uma empresa querem distribuir os seus trabalhadores pelas diversas tarefas por forma a minimizar o seu tempo de execução, maximizar a satisfação dos seus empregados quanto à tarefa que vão desempenhar e quanto ao horário de trabalho e ao grupo de colegas com que irão trabalhar; (3) os gestores de uma padaria pretendem fazer entregas de pão ao domicílio, minimizando o custo de transporte, a diferença entre o momento de entrega e o momento em que o cliente necessita do seu pão e o consumo de energia para a preparação do pão.

Quando procuramos resolver um problema multi-objectivo surgem algumas dificuldades. A solução de um problema de optimização multi-objectivo muito raramente é única, isto é, não existe uma solução que optimize em simultâneo o conjunto de todos os objectivos. Pelo contrário, existe um conjunto de alternativas, cada uma delas melhor (com valor inferior ou superior se pretendermos o mínimo ou máximo do objectivo, respectivamente), relativamente a alguns objectivos, mas pior (com valor superior ou inferior se pretendermos o mínimo ou máximo do objectivo, res-

pectivamente), relativamente a pelo menos um deles. Uma forma de abordar este tipo de problemas consiste em determinar todas as soluções que respeitam estas condições, designadas por soluções óptimas de Pareto ou soluções eficientes. Esta abordagem é interessante apenas do ponto de vista teórico, uma vez que o número destas soluções é, em geral, muito elevado e a simples enumeração do conjunto de todas estas soluções é uma tarefa muito difícil.

Pensou-se que, ao trabalhar no espaço dos objectivos em vez do espaço das variáveis de decisão, calculando as imagens das soluções eficientes, que designaremos por soluções não-dominadas, tornaria esta tarefa mais simples, isto porque, em geral, o espaço dos objectivos é de muito menor dimensão e muitas soluções eficientes têm a mesma imagem no espaço dos objectivos. Mas também o conjunto das soluções não-dominadas pode ter um número muito elevado de elementos e provou-se que isso pode acontecer mesmo para problemas com apenas dois objectivos (Ruhe, 1988a).

A escolha de uma alternativa ao cálculo de todas as soluções eficientes/não-dominadas aponta, tradicionalmente, para a utilização de uma função utilidade, construída com base nas preferências do utilizador. Obtida esta função, o problema a resolver consistiria na escolha de entre as alternativas encontradas daquela que maximizasse a função utilidade. Parece fácil, deste modo, chegar à melhor alternativa como solução do problema. A dificuldade surge aquando da construção da função utilidade. Essa dificuldade não está só em esperarmos obter uma função linear, mas também na impossibilidade da representação matemática dessa função em muitos casos (Steuer, 1986).

Outra ideia, para ultrapassar o problema do número elevado de soluções eficientes/não-dominadas, consiste na obtenção de uma representação deste conjunto de soluções. Na prática, parece ser esta uma das melhores formas de trabalhar com este tipo de problemas. No entanto, não deixa de ser importante a construção de algoritmos capazes de gerar todas aquelas soluções, pois a determinação de uma "boa" representação só será possível se pudermos fazer um estudo teórico do conjunto de todas estas soluções. Isso permitirá, por exemplo, medir a qualidade da representação.

O trabalho que será descrito nesta Tese foi iniciado com a leitura e exploração da literatura existente. Neste estudo inicial verificámos a existência de muitas falhas. Por exemplo: (1) apenas encontrámos propostos na literatura algoritmos para o problema de fluxos em redes com dois objectivos (ver Hamacher *et al.*, 2007b); (2) os modelos com variáveis contínuas eram escassos e muitos não resolviam os problemas que diziam resolver (ver Hamacher *et al.* 2007b, Lee & Pulat 1991, Malhotra & Puri 1984 e Pulat *et al.* 1992); (3) também para os modelos inteiros eram conhecidos alguns algoritmos, mas poucos determinavam os conjuntos que diziam determinar (ver Hamacher *et al.* 2007b, Huarng *et al.* 1992, Lee & Pulat 1993, Przybylski *et al.*

2006 e Sedeño Noda & González-Martín 2001), sendo o algoritmo Figueira (2000) o único capaz de determinar todas as soluções não-dominadas para o problema de fluxos inteiros em redes com dois objectivos ; (4) finalmente, relativamente aos métodos aproximativos, isto é, aos métodos que determinam apenas um subconjunto do conjunto das soluções eficientes/não-dominadas ou um conjunto para representar aquele, existiam também muito poucos trabalhos nesta área (ver Hamacher *et al.* 2007a,b, Mustafa & Goh 1998 e Nikolova 1998, 2001, 2003). Note-se que não estamos a ter em conta os métodos heurísticos e as meta-heurísticas para as aproximações.

Pareceu-nos, por isso, ser de grande interesse a investigação neste domínio. Os objectivos gerais definidos para esta Tese foram os seguintes: (1) fazer um estudo da bibliografia existente na área do problema de fluxos em redes; (2) explorar aspectos que ainda não tivessem sido completamente cobertos pela investigação até àquela data ou que se revelassem mal cobertos, com zonas ambíguas; (3) fazer uma caracterização matemática das soluções eficientes/não-dominadas suportadas para o problema de fluxos em redes inteiros com vários objectivos; (4) propor a resolução de algumas das questões por resolver.

Como objectivos de natureza experimental propusemos: (1) melhorar e implementar o algoritmo de Figueira (2000); (2) implementar os novos algoritmos propostos; (3) testar, através do uso de problemas gerados aleatoriamente, os algoritmos propostos.

Os resultados principais deste trabalho de investigação foram publicados em três artigos em revistas e mais alguns *working papers*:

- Eusébio, A., Figueira, J.R. e Ehrgott, M. 2008, *A primal-dual simplex algorithm for bi-objective network flow problems*, apresenta um novo algoritmo primal-dual para o cálculo das soluções não-dominadas extremas de um problema de fluxos em redes bi-objectivo;

- Eusébio, A. e Figueira, J.R. 2009, *Finding non-dominated solutions in bi-objective integer network flow problems*, descreve o algoritmo de restrição-$\varepsilon$, para o cálculo de todas as soluções eficientes/não-dominadas do problema de fluxos em redes inteiro bi-objectivo;

- Eusébio, A. e Figueira, J.R. 2009, *On the computation of all supported efficient solutions in multi-objective integer network flow problems*, além de provar a conexidade do conjunto das soluções suportadas num problema de fluxos em redes inteiro multi-objectivo, apresenta um novo algoritmo para as determinar;

- Eusébio, A. e Figueira, J.R. 2006, *Bi-criteria network flow problems: a characterization of non-dominated solutions*, onde se faz uma primeira caracteri-

zação das soluções não-dominadas para o problema de fluxos em redes inteiro bi-objectivo e onde se apresentam vários exemplos sobre este assunto;

- Eusébio, A. e Figueira, J.R. 2007, *A note on the computation of nondominated supported vectors in bicriteria network flow problems*, onde se apresenta uma primeira versão do algoritmo do circuito de custo zero para o problema de fluxos em redes inteiro bi-objectivo;

- Eusébio, A. e Figueira, J.R. 2008, *A negative-cycle algorithm for computing all the supported efficient solutions in multi-objective integer network flow problems*, onde se descreve o algoritmo do circuito de custo zero para um problema de fluxos em redes inteiro multi-objectivo, e se apresenta um exemplo e resultados computacionais da aplicação deste algoritmo.

O texto desta Tese foi dividido em três partes e oito capítulos, incluindo esta Introdução. Da Parte I fazem parte os capítulos 2 e 3, da Parte II os capítulos 4 e 5 e da Parte III os capítulos 6, 7 e 8.

Na Parte I, faremos uma revisão sobre os conceitos matemáticos de base e sobre o estado-da-arte para problemas de fluxos em redes multi-objectivo. Assim, no Capítulo 2, recordaremos a definição de conceitos sobre teoria de grafos, formularemos os diversos problemas a utilizar ao longo desta Tese, apresentaremos também de forma resumida o algoritmos simplex para redes e o algoritmo por eliminação dos circuitos de custo negativo para o problema de fluxo de custo mínimo numa rede. No Capítulo 3, faremos uma revisão sobre o estado-da-arte para problemas de fluxos em redes multi-objectivo, abordando os algoritmos que dizem determinar o conjunto de todas as soluções eficientes ou não-dominadas em problemas com variáveis contínuas e com variáveis inteiras e também os algoritmos aproximados.

Na Parte II, apresentaremos o trabalho realizado na área dos problemas lineares de fluxos em redes multi-objectivo com variáveis contínuas, nomeadamente, dois novos algoritmos do tipo primal-dual. No Capítulo 4, descreveremos um novo algoritmo primal-dual para o problema do fluxo de custo mínimo numa rede e, no Capítulo 5, apresentaremos um novo algoritmo primal-dual para o problema de fluxos em redes bi-objectivo em duas versões, a primeira versão como primal-dual simples e em segundo lugar uma versão melhorada daquele, aproveitando assim o conhecimento obtido ao longo da utilização do algoritmo.

Na Parte III, apresentaremos problemas lineares de fluxos em redes inteiros multi-objectivo onde descreveremos dois novos algoritmos e o melhoramento de um terceiro, assim como a sua implementação e resultados obtidos. No Capítulo 6, descreveremos um novo algoritmo para o cálculo de todas as soluções eficientes/não-dominadas suportadas de um problema multi-objectivo com variáveis inteiras, recor-

rendo ao uso de redes residuais e circuitos de custo zero. O algoritmo foi implementado e os resultados serão comentados. Apresentaremos também a demonstração da conexidade do conjunto destas soluções, com base no conceito de ciclo de custo zero. No Capítulo 7, apresentaremos uma versão melhorada do algoritmo de restrição-$\varepsilon$ (ver Figueira, 2000) para o cálculo do conjunto de todas as soluções não-dominadas. Descreveremos os melhoramentos e como foi feita a implementação deste algoritmo. Faremos, também, uma análise estatística dos resultados obtidos. Finalmente, no Capítulo 8 descreveremos um novo algoritmo para o cálculo de representações do conjunto de todas as soluções não-dominadas do problema de fluxos em redes inteiro bi-objectivo.

# Parte I

# Generalidades

Na primeira parte, abordamos os princípios gerais que servem de base para a elaboração desta Tese. Os conceitos matemáticos considerados ao longo do trabalho são aqui revistos de uma forma resumida e a notação de base é introduzida. No Capítulo 2, apresentamos o problema de fluxo de custo mínimo nas suas versões contínuo e inteiro e fazemos igualmente a revisão da sua formulação matemática. Recordamos também os diferentes tipos de condições de optimalidade para o problema de fluxo de custo mínimo, mais precisamente as condições de optimalidade dos custos reduzidos, dos desvios complementares e da inexistência de circuitos de custo negativo na rede residual. Fazem parte, também, deste primeiro capítulo os algoritmos simplex para redes e o algoritmo por eliminação dos circuitos de custo negativo. No final do primeiro capítulo são definidos os problemas de fluxos em redes multi-objectivo, ambas as versões com variáveis contínuas e variáveis inteiras, e ainda os conceitos de eficiência e dominância. No Capítulo 3, apresentamos o estado-da-arte sobre os problemas de fluxos em redes multi-objectivo. Este estudo foi dividido de acordo com o tipo de algoritmos em: a) algoritmos exactos, quando elaborados para calcular o conjunto de todas as soluções eficientes/não-dominadas; b) algoritmos aproximados, que são algoritmos com o objectivo de determinar apenas uma parte representativa daquele conjunto. Em cada uma destas secções são consideradas subsecções para problemas com variáveis contínuas e para variáveis inteiras. Finalmente, na última secção deste capítulo, fazemos uma análise crítica sobre os diversos métodos propostos para a determinação das soluções eficientes ou não-dominadas dos problemas de fluxos em redes com vários objectivos.

# Capítulo 2

# Conceitos, definições e notação de base

Este capítulo apresenta os conceitos, as respectivas definições e a notação de base que de alguma forma serão necessários nos próximos capítulos. Não é nosso objectivo tratar o conteúdo de uma forma pormenorizada, mas apenas relembrar certos conceitos e a notação utilizada. Para uma revisão mais detalhada dos assuntos abordados neste capítulo, podem ser consultadas, por exemplo, as seguintes obras: Ahuja *et al.* (1993), Bazaraa *et al.* (2005), Ehrgott (2005), Papadimitriou (1982) e Steuer (1986). Na primeira parte do capítulo, apresentamos conceitos da teoria de grafos e da optimização de fluxos em redes. Descrevemos e formulamos o problema do fluxo de custo mínimo, o seu dual e revemos alguns aspectos referentes à sua geometria em programação linear. Apresentamos os diferentes tipos de condições de optimalidade, as dos custos reduzidos, as dos desvios complementares e as da inexistência de circuitos de custo negativo. Fazemos, ainda, uma breve revisão dos métodos simplex e por eliminação dos circuitos de custo negativo para o problema de fluxos em redes com um único objectivo. Finalmente, os problemas de fluxos em redes multi-objectivo são descritos e são definidos os conceitos de dominância e eficiência.

## 2.1 Teoria de grafos e optimização de fluxos em redes

Nesta secção fazemos uma revisão sobre teoria de grafos e optimização de fluxos em redes e a geometria da programação linear. Revemos ainda o problema de fluxo de custo mínimo (para uma revisão detalhada sobre este assunto consultar a obra de Ahuja *et al.*, 1993).

### 2.1.1 Alguns conceitos de base da teoria de grafos

Consideremos um conjunto $\mathcal{V}$, não vazio, de elementos designados por *vértices*, *nodos* ou *nós* e um conjunto de pares de nodos, $\mathcal{A}$, não necessariamente ordenados. O par ordenado de conjuntos $(\mathcal{V}, \mathcal{A})$ é designado por *grafo*, $\mathcal{G} = (\mathcal{V}, \mathcal{A})$. Um elemento de $\mathcal{A}$ designa-se por *aresta*, representada por $\{i, j\}$, se a ordem dos nodos não é considerada, ou por *arco*, representado por $(i, j)$ ou $(j, i)$, se queremos dizer que o nodo $i$ precede o nodo $j$ ou que o nodo $j$ precede $i$, respectivamente. Deste modo, $(i, j)$ e $(j, i)$ representam arcos diferentes, enquanto $\{i, j\}$ e $\{j, i\}$ representam a mesma aresta. Numa aresta $\{i, j\}$ ou num arco $(i, j)$ ou $(j, i)$ dizemos que os nodos $i$ e $j$ são *adjacentes* em $\mathcal{G}$ e que esta aresta ou arco *liga* os nodos $i$ e $j$. Estes dizem-se extremidades da aresta ou do arco respectivo. No arco $(i, j)$ dizemos que $i$ é o *predecessor imediato* de $j$ e que $j$ é o *sucessor imediato* de $i$; $i$ é também designado de *extremidade inicial* ou *origem* e $j$ de *extremidade final* ou *destino*. Dizemos que dois nodos têm *arcos múltiplos* a ligá-los se são extremidades de dois ou mais arcos, em simultâneo. Assumimos, sem perda de generalidade, que os grafos considerados aqui não têm arcos múltiplos. Um grafo $\mathcal{G}$ diz-se *orientado* se todos os elementos de $\mathcal{A}$ são arcos. Um nodo diz-se *isolado* se não é extremidade de qualquer arco ou aresta. Um *lacete* é um arco em que as extremidades inicial e final coincidem. A todo o grafo pode associar-se um diagrama que, em geral, consiste em representar os nodos por círculos ou pontos e as arestas ou arcos por linhas, ligando os círculos ou pontos associados aos respectivos nodos. Na representação de um arco utiliza-se, habitualmente, uma seta para indicar a orientação do arco, da extremidade inicial para a final.

Consideremos os grafos $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ e $\mathcal{G}' = (\mathcal{V}', \mathcal{A}')$. $\mathcal{G}'$ diz-se um *subgrafo* de $\mathcal{G}$ se $\mathcal{V}' \subseteq \mathcal{V}$ e $\mathcal{A}' \subseteq \mathcal{A}$. Diz-se que $\mathcal{G}' = (\mathcal{V}', \mathcal{A}')$ é um *subgrafo* de $\mathcal{G}$ *gerado* por $\mathcal{V}'$ se $\mathcal{A}'$ contém todos os arcos de $\mathcal{A}$ com ambas as extremidades em $\mathcal{V}'$. Um grafo $\mathcal{G}' = (\mathcal{V}', \mathcal{A}')$ é um *subgrafo abrangente* ou *gerador* de $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ se $\mathcal{V}' = \mathcal{V}$ e $\mathcal{A}' \subseteq \mathcal{A}$.

Um *caminho*, num grafo orientado $\mathcal{G}$, é uma sequência, finita ou infinita, de arcos em que a extremidade final de um arco é a extremidade inicial do arco seguinte. Um caminho pode ser indicado pela sua sucessão de arcos, $(i_1, i_2)$-$(i_2, i_3)$-$\cdots$, ou pelos nodos que os ligam, $i_1$-$i_2$-$i_3$-$\cdots$. Num caminho finito, $(i_1, i_2)$-$(i_2, i_3)$-$\cdots$-$(i_{r-1}, i_r)$, $i_1$ e

$i_r$ dizem-se *extremidade inicial* e *extremidade final*, respectivamente, desse caminho. Um caminho finito em que as extremidades inicial e final correspondem ao mesmo nodo designa-se por *circuito*. Um *circuito elementar* é um circuito em que todos os seus nodos são distintos, com excepção do final que coincide com o inicial. O número de arcos de um caminho ou circuito designa-se por *comprimento* deste. Um caminho diz-se *caminho elementar* se todos os seus nodos são distintos. Um caminho diz-se *caminho simples* se todos os seus arcos são distintos; caso contrário, diz-se um *caminho composto*.

Uma *cadeia*, num grafo orientado, é uma sequência de nodos e arcos do grafo, $i_0$-$a_1$-$i_1$-$a_2$-$i_2$-$a_3$-$i_3$-$\cdots$ em que ou $a_k = (i_{k-1}, i_k)$ ou $a_k = (i_k, i_{k-1})$, $k = 1, 2, 3, \cdots$. Numa cadeia finita, $i_0$-$a_1$-$i_1$-$a_2$-$i_2$-$\cdots$-$i_{r-1}$-$a_r$-$i_r$, $i_1$ e $i_r$ dizem-se *extremidade inicial* e *extremidade final* da cadeia, respectivamente. Uma *cadeia elementar* não contém nodos repetidos. Uma *cadeia simples* não contém arcos repetidos. Um *ciclo* é uma cadeia finita ($r \geqslant 2$) em que as extremidades inicial e final correspondem ao mesmo nodo. Um *ciclo elementar* é um ciclo em que todos os seus nodos são distintos, com excepção da extremidade inicial que coincide com a final. Por vezes é necessário fixar o *sentido de um ciclo* como o sentido de um dos seus arcos $a_q$ e dizemos que este é de $i_{q-1}$ para $i_q$ ou de $i_q$ para $i_{q-1}$, consoante $a_q = (i_{q-1}, i_q)$ ou $a_q = (i_q, i_{q-1})$. No primeiro caso, dizemos que um arco do ciclo $a_k = (i_{k-1}, i_k)$ está orientado no sentido do ciclo. No segundo caso, dizemos que um arco do ciclo $a_k = (i_k, i_{k-1})$ está orientado no sentido do ciclo. Caso contrário, dizemos que o sentido do arco é contrário ao do ciclo. Um circuito é por vezes designado por *ciclo orientado*, uma vez que todos os arcos, no ciclo, têm o mesmo sentido. Nota: Nos capítulos seguintes utilizamos apenas circuitos e ciclos elementares que são designados por circuitos e ciclos, respectivamente.

Um grafo diz-se *conexo* se existir pelo menos uma cadeia elementar entre qualquer par de nodos. Um grafo diz-se *fortemente conexo* se todo o par de nodos estiver ligado por, pelo menos, um caminho elementar, em cada sentido. Um grafo conexo sem ciclos designa-se por *árvore*. Uma árvore $\mathcal{T} = (\mathcal{V}', \mathcal{A}')$ é uma *árvore geradora* de um grafo $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ se $\mathcal{T}$ é um subgrafo de $\mathcal{G}$ tal que $\mathcal{V}' = \mathcal{V}$ e $\mathcal{A}' \subseteq \mathcal{A}$.

Se a cada aresta e/ou nodo de um grafo associarmos um ou mais números reais estamos em presença de uma *rede* não orientada. Se o grafo é orientado diz-se que estamos em presença de uma rede *orientada*.

### 2.1.2 O problema do fluxo de custo mínimo

O problema do fluxo de custo mínimo (FCM) consiste na determinação de um fluxo como, por exemplo, a quantidade de um determinado produto, um certo número de objectos, dados, informação, etc., a transportar de determinados pontos de oferta

ou armazéns, de forma a satisfazer a procura situada noutros pontos (nodos) da rede, ao menor custo total possível. O fluxo em cada arco $(i, j) \in \mathcal{A}$ é designado por $x_{ij}$. A rede subjacente ao problema FCM é definida da seguinte forma:

Consideremos um grafo orientado $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, onde $\mathcal{V} = \{1, 2, \cdots, m\}$ é o conjunto de nodos e $\mathcal{A} = \Big\{(i_1, j_1),\ (i_2, j_2),\ \cdots,\ (i_n, j_n)\Big\}$ é o conjunto de arcos de $\mathcal{G}$. Cada arco tem associados um *custo unitário* $c_{ij}$, um *limite inferior* de fluxo $l_{ij}$ e um *limite superior* de fluxo $u_{ij}$, onde $u_{ij} \geqslant l_{ij}$ (ao limite superior também costumamos chamar capacidade do arco). O número de nodos e o número de arcos na rede são designados por $m$ e $n$, respectivamente. Cada nodo $k \in \mathcal{V}$ pode ter uma oferta ou procura, $b_k$, associada. Se $b_k > 0$, o nodo designa-se por *nodo de oferta*; se $b_k < 0$, o nodo designa-se por *nodo de procura*. Um nodo em que $b_k = 0$ designa-se por *nodo de transbordo* ou *ponto de passagem* de fluxo.

Uma rede é representada por $\mathcal{N} = (\mathcal{G},\ c,\ l,\ u,\ b)$, onde $\mathcal{G} = (\mathcal{V}\ ,\mathcal{A})$ é um grafo orientado, $c = (c_{i_1j_1},\ c_{i_2j_2},\ \cdots,\ c_{i_nj_n}) \in \mathbb{Z}^n$ é o vector de "custos", $l = (l_{i_1j_1},\ l_{i_2j_2},\ \cdots,\ l_{i_nj_n}) \in \mathbb{Z}^n$ é o vector dos limites inferiores do fluxo, $u = (u_{i_1j_1},\ u_{i_2j_2},\ \cdots,\ u_{i_nj_n}) \in \mathbb{Z}^n$ é o vector dos limites superiores do fluxo e $b = (b_1,\ b_2,\ \cdots,\ b_m) \in \mathbb{Z}^m$ é o vector das ofertas/procuras. A Figura 2.1 apresenta a rede $\mathcal{N}$, exemplo retirado de Pulat *et al.* (1992), onde

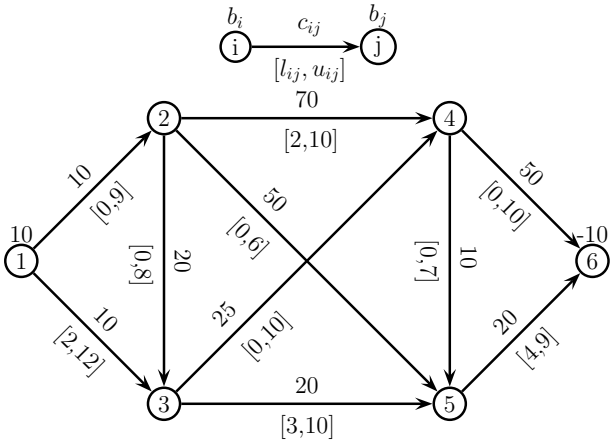



**Figura 2.1:** *Representação gráfica de uma rede.*

- $\mathcal{V} = \{1, 2, 3, 4, 5, 6\}$;
- $\mathcal{A} = \{(1, 2), (1, 3), (2, 3), (2, 4), (2, 5), (3, 4), (3, 5), (4, 5), (4, 6), (5, 6)\}$

- $c = (10, 10, 20, 70, 50, 25, 20, 10, 50, 20)$
- $l = (0, 2, 0, 2, 0, 0, 3, 0, 0, 4)$
- $u = (9, 12, 8, 10, 6, 10, 10, 7, 10, 9)$
- $b = (10, 0, 0, 0, 0, -10)$

Numa rede $\mathcal{N}$ podemos definir o problema FCM que consiste em identificar a quantidade de fluxo, a passar em cada arco $(i, j)$, $l_{ij} \leqslant x_{ij} \leqslant u_{ij}$, de forma a satisfazer a procura a partir da oferta disponível, minimizando o custo total desta operação de transporte do fluxo ao longo dos arcos de $\mathcal{G}$.

### 2.1.3 Rede residual

Na construção e implementação de algoritmos para a resolução do problema FCM pode ser conveniente (mesmo que de forma indirecta) utilizar a capacidade do fluxo que resta em cada arco. Assim, a rede inicial é modificada dando origem à denominada *rede residual*, que definimos a seguir.

Consideremos a rede $\mathcal{N}$ associada a um problema FCM e um fluxo $x = (x_{i_1j_1}, x_{i_2j_2}, \cdots, x_{i_nj_n})$ nesta rede. A *rede residual*, $\mathcal{N}(x)$, relativamente ao fluxo dado $x$, é a rede que resulta de $\mathcal{N}$ substituindo cada arco $(i, j) \in \mathcal{A}$ pelos dois arcos $(i, j)$ e $(j, i)$: o arco $(i, j)$ tem custo $c_{ij}$ e uma *capacidade residual* de $r_{ij} = u_{ij} - x_{ij}$ e o arco $(j, i)$ tem custo $-c_{ij}$ e capacidade residual $r_{ji} = x_{ij} - l_{ij}$. A rede residual é formada apenas pelos arcos com capacidade residual positiva. Pode mostrar-se que todo o fluxo $x$ na rede $\mathcal{N}$ corresponde a um fluxo $x'$ numa rede residual associada (Ahuja *et al.*, 1993).

A soma $\sum \delta_{ij} c_{ij}$ para todos os arcos $(i, j)$ num ciclo designa-se por *custo do ciclo*, onde $\delta_{ij}$ é igual a 1 se o arco $(i, j)$ tiver o sentido do ciclo e $\delta_{ij}$ é igual a $-1$ se o arco $(i, j)$ tem sentido contrário ao do ciclo. Um ciclo (não necessariamente orientado) em $\mathcal{N}$ é designado por *ciclo incremental* relativamente ao fluxo $x$, se ao adicionarmos uma quantidade positiva de fluxo ao ciclo (adicionando ou subtraindo essa quantidade ao fluxo dos arcos do ciclo consoante o seu sentido é ou não o do ciclo), o fluxo permanece admissível (ver definição de fluxo admissível na Subsecção 2.2.1). Deste modo, um ciclo incremental não pode conter arcos $(i, j)$ de sentido contrário ao do ciclo tais que $x_{ij} = l_{ij}$ ou arcos com o sentido do ciclo tais que $x_{ij} = u_{ij}$. Cada ciclo incremental relativamente a $x$ corresponde a um circuito na rede residual $\mathcal{N}(x)$ e *vice-versa* (ver Ahuja *et al.*, 1993 e Bazaraa *et al.*, 2005).

Como exemplo de uma rede residual, consideremos a rede da Figura 2.1 e o fluxo nesta rede $x = (x_{12}, x_{13}, x_{23}, x_{24}, x_{25}, x_{34}, x_{35}, x_{45}, x_{46}, x_{56}) = (2, 8, 0, 2, 0, 0, 8, 1, 1, 9)$.

A rede residual $\mathcal{N}(x)$ está representada na Figura 2.2. Por exemplo, ao arco $(1,2)$, na rede inicial, com fluxo 2 estão associados, na rede residual, o arco $(1,2)$, com capacidade residual $r_{12} = u_{12} - x_{12} = 7$ e custo $c_{12} = 10$ e o arco $(2,1)$, com $r_{21} = x_{12} = 2$ e custo $-c_{12} = -10$. Mas relativamente ao arco $(2,3)$, na rede inicial, está associado apenas o arco $(2,3)$, na rede residual, com $r_{23} = 8$ e custo $c_{23} = 20$.

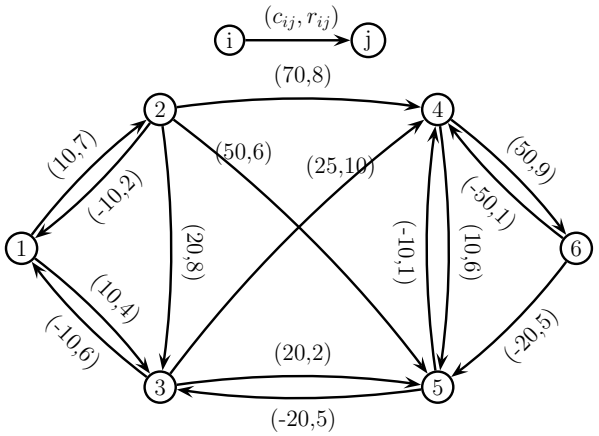



**Figura 2.2:** *Rede residual associada à rede da Figura 2.1 e à solução* $x = (2,\ 8,\ 0,\ 2,\ 0,\ 0,\ 8,\ 1,\ 1,\ 9)$.

## 2.2 O problema do fluxo de custo mínimo enquanto problema de programação linear

O problema FCM pode ser formulado como um problema de programação linear. Apresentamos a seguir a formulação primal e dual, para este problema. Admitimos, sem perda de generalidade, que o limite inferior para o fluxo no arco $(i,j)$ é zero, isto é, $l_{ij} = 0$, $\forall (i,j) \in \mathcal{A}$. Caso $l_{ij}$ seja diferente de zero, podemos converter o problema num outro com $l'_{ij} = 0$, bastando para isso fazer a mudança de variável $x_{ij}$ por $x'_{ij} = x_{ij} - l_{ij}$. Os limites da nova variável serão $l'_{ij} = 0$ e $u'_{ij} = u_{ij} - l_{ij}$. O novo problema tem o mesmo número de soluções e as soluções de um podem ser obtidas a partir das do outro, através de uma simples transformação linear.

### 2.2.1 O problema primal

Consideremos a rede $\mathcal{N} = (\mathcal{G}, c, l, u, b)$. Designamos por *fluxo admissível* na rede uma função que a cada arco $(i, j)$ em $\mathcal{A}$ associa o número real $x_{ij}$ satisfazendo as seguintes restrições:

- as *restrições da conservação do fluxo* ou equações de Kirchhoff, que dizem que a quantidade de fluxo que entra num nodo menos a quantidade de fluxo que sai é igual à procura/oferta de fluxo no nodo:

$$\sum_{(k,j)\in\mathcal{A}} x_{kj} - \sum_{(i,k)\in\mathcal{A}} x_{ik} = b_k, \quad \forall k \in \mathcal{V} \quad \text{e}$$

- as *restrições de capacidade*, que dizem que o fluxo num arco não pode estar fora dos limites de capacidade desse arco:

$$0 \leqslant x_{ij} \leqslant u_{ij}, \forall (i,j) \in \mathcal{A}.$$

Representamos este fluxo por $x = (x_{i_1j_1}, x_{i_2j_2}, \cdots, x_{i_nj_n})$. O *custo do fluxo* $x$ é a soma $\sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij}$. Um fluxo de custo mínimo é um fluxo com o menor custo, entre todos os fluxos existentes na rede $\mathcal{N}$.

O problema FCM associado à rede $\mathcal{N}$, pode ser formulado, matematicamente, da seguinte forma:

$$\text{minimizar} \quad \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij}, \tag{2.1}$$

$$\text{sujeito a:} \quad \sum_{(k,j)\in\mathcal{A}} x_{kj} - \sum_{(i,k)\in\mathcal{A}} x_{ik} = b_k, \quad \forall k \in \mathcal{V}, \tag{2.1.a}$$

$$0 \leqslant x_{ij} \leqslant u_{ij}, \forall (i,j) \in \mathcal{A}, \tag{2.1.b}$$

ou de forma mais compacta:

$$\begin{array}{ll} \text{minimizar} & c^T x, \\ \text{sujeito a:} & x \in X, \end{array} \tag{2.2}$$

onde

$$X = \Big\{ x = (x_{i_1j_1}, \cdots, x_{i_nj_n}) \in \mathbb{R}^n : \begin{array}{l} \displaystyle\sum_{(k,j)\in\mathcal{A}} x_{kj} - \sum_{(i,k)\in\mathcal{A}} x_{ik} = b_k, \ \forall k \in \mathcal{V}, \\ 0 \leqslant x_{ij} \leqslant u_{ij}, \forall (i,j) \in \mathcal{A} \Big\} \end{array} \tag{2.3}$$

é a *região admissível* no espaço das variáveis de decisão. Ao longo desta Tese, a designação região admissível será utilizada sem especificar o espaço respectivo quando nos referirmos ao espaço das variáveis de decisão.

### 2.2.2 O problema dual

A formulação (2.1) é conhecida como *problema primal*. A resolução do problema FCM faz-se geralmente com recurso ao problema associado, conhecido por *problema dual*.

Para construirmos o problema dual de (2.1) comecemos por associar a cada uma das restrições deste problema uma variável. Designemos por $\pi_k$ a variável dual associada à restrição da conservação do fluxo correspondente ao nodo $k$, isto é, a variável dual associada à restrição de ordem $k$ em (2.1.a), para todo o $k$ em $\mathcal{V}$. Designemos por $\mu_{ij}$ a variável dual associada à restrição de capacidade do arco $(i,j)$, isto é, à restrição $x_{ij} \leqslant u_{ij} \Leftrightarrow -x_{ij} \geqslant -u_{ij}$ em (2.1.b), para todo o arco $(i,j)$ em $\mathcal{A}$. O problema dual pode ser então formulado da seguinte forma:

$$\begin{array}{rlrcl} \text{maximizar} & \displaystyle\sum_{k\in\mathcal{V}} b_k\pi_k - \sum_{(i,j)\in\mathcal{A}} u_{ij}\mu_{ij}, & & & \\ \text{sujeito a:} & \pi_i - \pi_j - \mu_{ij} & \leqslant & c_{ij} & , \forall (i,j) \in \mathcal{A}, \\ & \mu_{ij} & \geqslant & 0 & , \forall (i,j) \in \mathcal{A}. \end{array} \tag{2.4}$$

Por hipótese os limites superiores dos arcos $(i,j)$, $\forall (i,j) \in \mathcal{A}$, são não negativos, isto é, $u_{ij} \geqslant 0, \forall (i,j) \in \mathcal{A}$.

O *custo reduzido* do arco $(i,j), \forall (i,j) \in \mathcal{A}$, é definido da seguinte forma:

$$\bar{c}_{ij} = c_{ij} - \pi_i + \pi_j. \tag{2.5}$$

### 2.2.3 Estrutura de árvore geradora

No problema FCM a matriz dos coeficientes das variáveis nas restrições (2.1.a) é designada por *matriz de incidência nodo-arco*. Esta matriz tem $m$ linhas e $n$ colunas. Cada coluna está associada a um arco $(i,j) \in \mathcal{A}$ e tem todas as suas componentes nulas, excepto as das linhas $i$ e $j$, que são 1 e $-1$, respectivamente. Pode provar-se que a característica desta matriz é $m-1$ (ver Bazaraa *et al.*, 2005). O problema pode ser modificado adicionando uma variável artificial $x_{01}$ de tal forma que a matriz, $A$, resultante tenha característica $m$. Acrescentar esta variável ao problema corresponde a adicionarmos um arco artificial à rede. Este arco é designado por *arco raiz*. O novo conjunto de restrições pode então ser escrito, na forma matricial, como $Ax = b$, onde $x = (x_{01}, x_{i_1j_1}, x_{i_2j_2}, \cdots, x_{i_nj_n})$ é o vector das variáveis e $b = (b_1, b_2, \cdots, b_m)$ é o vector das ofertas/procuras. Designemos por $B$ uma submatriz não singular de $A$ e suponhamos, sem perda de generalidade, que as colunas de $B$ são as primeiras $m$ colunas de $A$. A submatriz restante $N$ é dividida em duas submatrizes $N_L$ e $N_U$. Suponhamos, sem perda de generalidade, que $N = [N_L \; N_U]$. Assim, a matriz

$A$ pode ser dividida em três submatrizes, $A = [B\ N_L\ N_U]$ e o vector $x$ em três subvectores, $x = (x_B, x_L, x_U)$, de tal forma que

$$Ax = b \Leftrightarrow [B\ N_L\ N_U] \begin{bmatrix} x_B \\ x_L \\ x_U \end{bmatrix} = b.$$

Deste modo, podemos reescrever o novo sistema de restrições:

$$Ax = b \Leftrightarrow x_B = B^{-1}b - B^{-1}N_L x_L - B^{-1}N_U x_U. \tag{2.6}$$

Designemos por $\mathcal{T}$, $L$ e $U$ o conjunto de arcos associados às colunas das matrizes $B$, $N_L$ e $N_U$, respectivamente. Eventualmente, $L$ ou $U$ podem ser conjuntos vazios. Sendo $\mathcal{V}$ o conjunto de nodos da rede dada, o grafo $(\mathcal{V}, \mathcal{T})$ é uma árvore geradora de $\mathcal{G}$ (Ahuja *et al.*, 1993) designada por *árvore geradora com raiz*. Uma solução $x$ tal que $x_{ij} = 0, \forall (i,j) \in L$ e $x_{ij} = u_{ij}, \forall (i,j) \in U$ é designada por *solução básica* e o terno $(\mathcal{T}, L, U)$ é designado por *estrutura de árvore geradora* $(EAG)$. Note que o conhecimento dos arcos que estão em $L$ e dos arcos que estão em $U$ é suficiente para identificarmos a solução básica associada e, por isso, utilizamos a notação $\rho_\nu = (L, U)$ para a representar, onde $\nu$ designa um número que identifica a respectiva solução. Note que conhecidos os conjuntos $L$ e $U$ conhecemos igualmente $x_L$ e $x_U$ em (2.6) e consequentemente também $x_B$. Designamos os arcos em $\mathcal{T}$ por *arcos básicos* e aqueles que não pertencerem a $\mathcal{T}$ por *arcos não-básicos*. Uma solução $x$ satisfazendo (2.6), tal que $0 \leqslant x_{ij} \leqslant u_{ij}, \forall (i,j) \in \mathcal{A}$, designa-se por *solução admissível* e $(\mathcal{T}, L, U)$ por *EAG admissível*. Uma $EAG$ diz-se *degenerada* se existir um arco $(i,j)$ em $\mathcal{T}$ tal que $x_{ij} = 0$ ou $x_{ij} = u_{ij}$. Neste caso, diferentes $EAG$'s degeneradas podem estar associadas a uma mesma solução. De facto, pode ser possível obter uma nova $EAG$ a partir da inicial, trocando um arco $(i,j)$ em $\mathcal{T}$, tal que $x_{ij} = 0$ ou $x_{ij} = u_{ij}$, por um arco em $L$ ou $U$, respectivamente. Se uma solução, $\nu$, tem $r > 1$ $EAG$s degeneradas associadas, representamos cada uma delas por $\rho_\nu^k$, $k = 1, 2, \cdots, r$.

## 2.3 O problema inteiro do fluxo de custo mínimo

Na Secção 2.2 não colocamos qualquer restrição ao tipo de "produto" a ser transportado na rede, admitindo, deste modo, a possibilidade de dividir esse "produto" em pequenas quantidades, se necessário. Acontece, no entanto, que em muitos dos problemas FCM práticos o "produto" na rede corresponde a objectos indivisíveis devendo, por isso, o fluxo na rede ser inteiro. Como exemplo, podemos referir a distribuição de automóveis de uma determinada marca dos locais de fabrico para os

diversos concessionários. Passamos então a ter o problema do fluxo inteiro de custo mínimo (FICM) que, na sua formulação, contém a restrição adicional:

$$x_{ij} \text{ inteiro}, \forall (i,j) \in \mathcal{A}$$

Assim o problema FICM pode ser formulado da seguinte forma:

$$\begin{array}{ll} \text{minimizar} & c^T x, \\ \text{sujeito a:} & x \in X^I, \end{array} \tag{2.7}$$

onde

$$X^I = \Big\{ x = (x_{i_1 j_1}, \cdots, x_{i_n j_n}) \in \mathbb{N}_0^n : \sum_{(k,j)\in\mathcal{A}} x_{kj} - \sum_{(i,k)\in\mathcal{A}} x_{ik} = b_k, \ \forall k \in \mathcal{V} \\ x_{ij} \leqslant u_{ij}, \forall (i,j) \in \mathcal{A} \Big\}. \tag{2.8}$$

## 2.4 Geometria da programação linear

Nesta secção fazemos uma revisão sobre a representação geométrica da região admissível de um problema de programação linear (ver Papadimitriou, 1982, Schrijver, 2000 e Steuer, 1986).

As restrições do problema de programação linear (2.1) podem ser vistas como restrições do tipo $a_1 x_1 + a_2 x_2 + \cdots + a_n x_n = b$, onde $a_1, a_2, \cdots, a_n$ e $b$ são números reais. O conjunto dos pontos $x = (x_1, x_2, \cdots, x_n) \in \mathbb{R}^n$ que satisfazem esta restrição designa-se por *hiperplano*, isto é, um hiperplano $H$ é o conjunto

$$H = \{x = (x_1, x_2, \cdots, x_n) \in \mathbb{R}^n : a_1 x_1 + a_2 x_2 + \cdots + a_n x_n = b, \ b \in \mathbb{R}, \\ a_i \in \mathbb{R}, i = 1, 2, \cdots, n\},$$

onde $a_i \neq 0$ para pelo menos um dos $i = 1, 2, \cdots, n$. Um hiperplano define dois semi-espaços (fechados):

$$SE^- = \{x \in \mathbb{R}^n : a_1 x_1 + a_2 x_2 + \cdots + a_n x_n \leqslant b\}$$

e

$$SE^+ = \{x \in \mathbb{R}^n : a_1 x_1 + a_2 x_2 + \cdots + a_n x_n \geqslant b\}.$$

A intersecção de um número finito de semi-espaços, quando limitada e não vazia, designa-se por *poliedro*. Pode mostrar-se facilmente que a região admissível de um problema FCM (2.1), $X$ não vazio, é um poliedro. O conjunto $X$ é também um *conjunto convexo*, isto é, $\forall x', x'' \in X$ e $\forall \lambda \in [0,1]$, $\lambda x' + (1-\lambda) x'' \in X$.

Consideremos um hiperplano $H$ e designemos por $d$ a dimensão do poliedro $X$ e $SE$ um dos semi-espaços definido por $H$. Se $F = X \cap SE$ é um subconjunto de $H$ então $F$ designa-se por *face* de $X$ e $H$ por hiperplano de suporte que define $F$. Um *vértice*, uma *aresta* e uma *faceta* são faces de dimensão zero, um e $d-1$, respectivamente. Pode mostrar-se que as coordenadas de qualquer ponto num poliedro $X$ podem ser escritas como uma combinação convexa das coordenadas dos seus vértices ($d+1$ vértices são suficientes para escrever qualquer ponto). Uma face $F \subseteq X$ de dimensão $r$ é uma *face máxima* se não existir outra face $F' \subseteq X$ de dimensão $r'$ tal que $F \subset F'$ e $r < r'$.

Consideremos o conjunto das restrições (2.1.a) e (2.1.b).

$$\begin{array}{l} Ax = b, \\ 0 \leqq x \leqq u. \end{array} \tag{2.9}$$

De (2.6) temos, $x_B = B^{-1}b - B^{-1}N_L x_L - B^{-1}N_U x_U$. Assim (2.9) é equivalente a

$$\begin{array}{l} 0 \leqq B^{-1}b - B^{-1}N_L x_L - B^{-1}N_U x_U \leqq u_B, \\ 0 \leqq x_L \leqq u_L, \\ 0 \leqq x_U \leqq u_U, \end{array} \tag{2.10}$$

onde $u_L$ e $u_U$ são os vectores formados pelas componentes de $u$ correspondentes às variáveis $x_B$, $x_L$ e $x_U$, respectivamente. O conjunto de restrições (2.10) descreve a intersecção (limitada e não vazia) de semi-espaços e, por isso, define um poliedro $P$. Pode mostrar-se que $P \subset \mathbb{R}^{n+1-m}$ e que qualquer vértice de $P$ corresponde a uma $EAG$ admissível do problema (2.1) e *vice-versa*. Sabemos também que, num poliedro, qualquer vértice é um ponto extremo e *vice-versa*. Desta forma, podemos dizer que a qualquer ponto extremo está associada uma $EAG$ admissível do problema e a cada $EAG$ admissível está associado um ponto extremo do poliedro $P$.

## 2.5 Os diferentes tipos de condições de optimalidade

As condições de optimalidade aparecem como um método para averiguarmos se uma determinada solução é óptima ou não. O seu conhecimento sugere a construção de algoritmos para a determinação da solução óptima (ou soluções óptimas), para o problema respectivo.

Sabemos que quando um problema de programação linear tem uma região admissível limitada, como é o caso do problema FCM considerado e uma única solução óptima, então essa solução é um ponto extremo da sua região admissível. Sabemos também que qualquer ponto da região admissível é um ponto extremo se e só se é uma solução básica admissível. Quando o problema tem mais que uma solução

óptima, qualquer uma pode ser escrita como combinação convexa dos seus pontos extremos óptimos e qualquer combinação convexa destes é uma solução óptima para o problema. Desta forma, o objectivo principal na resolução deste tipo de problemas é o de encontrar os pontos extremos óptimos.

No problema FCM podem ser considerados, em geral, três tipos de condições de optimalidade: dos custos reduzidos, dos desvios complementares e da inexistência de circuitos de custo negativo na rede residual.

### 2.5.1 Custos reduzidos

Uma forma bastante usual de verificarmos a optimalidade de uma determinada solução é através do sinal dos custos reduzidos (2.5). O algoritmo simplex para redes é um dos algoritmos que utiliza esta condição de optimalidade. Como vimos antes, toda a solução básica está associada com uma estrutura $EAG$ na rede. Pode mostrar-se que o custo reduzido dos arcos básicos é sempre igual a zero e, como tal, o seu valor nada nos diz sobre a optimalidade ou não de uma solução. É através do sinal dos custos reduzidos dos arcos não-básicos que a optimalidade será averiguada, de acordo com o teorema seguinte.

**Teorema 2.5.1** *Designemos por $x^*$ uma solução básica admissível para o problema FCM e por $(\mathcal{T}, L, U)$ a EAG associada.*

*a) Se $\bar{c}_{ij} \geqslant 0, \forall (i,j) \in L$ e $\bar{c}_{ij} \leqslant 0, \forall (i,j) \in U$, então $x^*$ é óptima.*

*b) Se $x^*$ é óptima e não degenerada, então $\bar{c}_{ij} \geqslant 0, \forall (i,j) \in L$ e $\bar{c}_{ij} \leqslant 0, \forall (i,j) \in U$.*

Uma $EAG$ nas condições da alínea a) do Teorema 2.5.1, isto é, tal que $\bar{c}_{ij} \geqslant 0, \forall (i,j) \in L$ e $\bar{c}_{ij} \leqslant 0, \forall (i,j) \in U$ designa-se por *EAG óptima*.

### 2.5.2 Desvios complementares

Dada uma solução para o problema primal e uma solução para o problema dual, podemos averiguar se estas soluções são simultaneamente óptimas, para os respectivos problemas, utilizando as denominadas relações dos desvios complementares. Consideremos o problema FCM (2.1) e o seu dual (2.4). Designemos por $x^*$ e $(\pi^*, \mu^*)$ duas soluções admissíveis para o primeiro e segundo problemas, respectivamente. Estas soluções são ambas óptimas para os problemas respectivos se e só se tiverem o mesmo valor objectivo, isto é, se

$$\sum_{(i,j)\in\mathcal{A}} c_{ij} x^*_{ij} = \sum_{k\in\mathcal{V}} b_k \pi^*_k - \sum_{(i,j)\in\mathcal{A}} u_{ij} \mu^*_{ij}.$$

**Teorema 2.5.2 (Teorema dos desvios complementares)** *Consideremos as soluções admissíveis $x^*$ e $(\pi^*, \mu^*)$ para os problemas primal (2.1) e dual (2.4), respectivamente. Então $x^*$ e $(\pi^*, \mu^*)$ são óptimas se e só se*

$$\begin{aligned} x_{ij}^* \left(c_{ij} - \pi_i^* + \pi_j^* + \mu_{ij}^*\right) &= 0, \forall (i,j) \in \mathcal{A} \text{ e} \\ \mu_{ij}^*(x_{ij}^* - u_{ij}) &= 0, \forall (i,j) \in \mathcal{A}. \end{aligned}$$

Este teorema diz-nos que pelo menos um dos factores em cada um dos produtos deve ser zero. Em particular

$$\begin{array}{rcrcrcl} x_{ij}^* & < & u_{ij} & \Rightarrow & \mu_{ij}^* & = & 0, \\ \mu_{ij}^* & > & 0 & \Rightarrow & x_{ij}^* & = & u_{ij}. \end{array}$$

Podemos deduzir outras implicações deste teorema. Se o custo reduzido do arco $(i,j)$ não é igual a zero então o valor do fluxo no arco ou é igual a 0 ou é igual a $u_{ij}$. Mais precisamente,

$$\pi_i^* - \pi_j^* < c_{ij} \Rightarrow x_{ij}^* = 0 \text{ e} \tag{2.11}$$

$$\pi_i^* - \pi_j^* > c_{ij} \Rightarrow x_{ij}^* = u_{ij}. \tag{2.12}$$

De facto, para a primeira condição temos,

$$\begin{aligned} \pi_i^* - \pi_j^* < c_{ij} &\Rightarrow c_{ij} - \pi_i^* + \pi_j^* > 0 \\ &\Rightarrow c_{ij} - \pi_i^* + \pi_j^* + \mu_{ij}^* > 0 \\ &\Rightarrow x_{ij}^* = 0 \end{aligned}$$

e para a segunda

$$\begin{aligned} \pi_i^* - \pi_j^* > c_{ij} &\Rightarrow c_{ij} - \pi_i^* + \pi_j^* < 0 \\ &\Rightarrow \mu_{ij}^* > 0 \\ &\Rightarrow x_{ij}^* = u_{ij}. \end{aligned}$$

Podemos ainda dizer que o custo reduzido de um arco é zero, sempre que o seu fluxo não é igual a um dos limites, isto é,

$$0 < x_{ij}^* < u_{ij} \Rightarrow \pi_i^* - \pi_j^* = c_{ij}.$$

### 2.5.3 Inexistência de circuito de custo negativo

Consideremos agora a rede residual associada ao problema FCM. A optimalidade de uma solução básica admissível, $x^*$, para o problema FCM pode ser avaliada através do custo dos circuitos numa rede residual. Sabe-se que uma solução admissível $x^*$ é óptima se a rede residual $\mathcal{N}(x^*)$ não contém qualquer circuito de custo negativo, de acordo com o teorema seguinte (ver Ahuja *et al.*, 1993).

**Teorema 2.5.3** *Designemos por $x^*$ uma solução admissível para o problema FCM. A solução $x^*$ é óptima para este problema se e só se a rede residual $\mathcal{N}(x^*)$ não contém qualquer circuito de custo negativo.*

As condições de optimalidade do custo reduzido são aplicáveis também à rede residual com as alterações ajustadas à nova rede. Estas condições são enunciadas pelo teorema seguinte.

**Teorema 2.5.4** *Designemos por $x^*$ uma solução básica admissível para o problema FCM e seja $(\mathcal{T}, L, U)$ a EAG associada a esta solução. A solução $x^*$ é óptima para este problema se e só se os custos reduzidos são não negativos para todo o arco na rede $\mathcal{N}(x^*)$, isto é,*

$$\bar{c}_{ij} \geqslant 0, \text{ para todo o arco } (i,j) \text{ em } \mathcal{N}(x^*).$$

Designemos por $W$ um circuito na rede residual. Sabemos que $\sum_{(i,j)\in W} \bar{c}_{ij} = \sum_{(i,j)\in W} c_{ij}$. Esta igualdade permite estabelecer a equivalência entre as condições de optimalidade da inexistência de circuito de custo negativo e as condições de optimalidade dos custos reduzidos.

## 2.6 Resolução do problema do fluxo de custo mínimo

Nesta secção apresentamos dois algoritmos frequentemente utilizados para a resolução do problema FCM. O primeiro é o algoritmo simplex para redes. Este algoritmo difere do simplex para problemas de programação linear pela estrutura de dados utilizada, pois, em vez de utilizar matrizes o método simplex para redes, utiliza uma representação em estrutura de árvore geradora. Este facto conduz a uma poupança de recursos computacionais, traduzida numa menor necessidade de memória e numa redução do número de operações necessárias para a determinação de uma solução óptima de um problema FCM. O segundo algoritmo é um algoritmo por eliminação dos circuitos negativos. Este algoritmo procura a solução cuja rede residual associada não tem qualquer circuito de custo negativo.

Na resolução do problema vamos admitir que o grafo $\mathcal{G}$ associado à rede $\mathcal{N}$ definida na Subsecção 2.1.2 é conexo. Só desta forma poderemos utilizar o método simplex para redes. Quando o grafo não é conexo nem sempre o problema é não admissível, mas neste caso é possível dividir o problema inicial em subproblemas, associados com subgrafos conexos e resolvê-los separadamente.

Podemos, na resolução do problema do fluxo de custo mínimo, admitir que o total da oferta é igual ao total da procura, isto é, que $\sum_{k\in\mathcal{V}} b_k = 0$. De facto, se a procura for superior à oferta, o problema é não admissível, uma vez que não existe recurso suficiente para satisfazer a procura. Se a oferta for superior à procura podemos modificar a rede inicial, acrescentando mais um ponto de procura capaz de receber toda a oferta em excesso a um custo nulo, isto é, acrescentar um nodo $m+1$, com $b_{m+1} = -\sum_{k\in\mathcal{V}} b_k$ e arcos $(i, m+1), \forall i \in \mathcal{V}$ com custo $c_{i\,m+1} = 0$. A solução do problema na rede modificada dá-nos a solução do problema inicial apagando o nodo e arcos acrescentados.

### 2.6.1 Método simplex para redes

Recordemos sucintamente o método simplex para redes, para o problema FCM (Algoritmo 1). Esta revisão é feita com base em Figueira (2000). Consideremos o problema FCM definido num grafo orientado $\mathcal{G}$ com um arco raiz. A ideia de base do método simplex para redes consiste na construção de uma $EAG$ da forma $(\mathcal{T}, L, U)$. Nesta estrutura, como visto na Secção 2.2.3, $m$ arcos pertencem a $\mathcal{T}$ e os restantes pertencem a $L$ e/ou $U$ consoante o valor do fluxo no arco respectivo é 0 ou $u_{ij}$, respectivamente. O problema FCM, quando admissível, tem sempre pelo menos uma $EAG$ óptima (ver Ahuja *et al.*, 1993). É possível encontrar uma $EAG$ óptima passando de uma $EAG$ admissível para outra adjacente, sucessivamente. Em cada iteração, trocamos um par de arcos (um arco que entra em $\mathcal{T}$ e outro que sai de $\mathcal{T}$, que poderá, eventualmente ser o mesmo arco) através de uma operação de *pivotação*, que consiste em:

(1) Identificar um arco elegível $(k, l)$ com $(k, l) \notin \mathcal{T}$, isto é, um arco que não satisfaça as condições de optimalidade, ou seja, um arco, $(k, l)$, não pertencente a $\mathcal{T}$ tal que:

   *i*) o seu custo reduzido , $\bar{c}_{kl}$, é estritamente negativo e o seu fluxo está no seu limite inferior, isto é, $\bar{c}_{kl} < 0$ e $(k, l) \in L$, ou

   *ii*) o seu custo reduzido é estritamente positivo e o seu fluxo está no seu limite superior, isto é, $\bar{c}_{kl} > 0$ e $(k, l) \in U$.

(2) Adicionar o arco $(k, l)$ a $\mathcal{T}$ e determinar um arco $(p, q)$ que sai de $\mathcal{T}$.

(3) Determinar a nova $EAG$ e as respectivas soluções primal, $x$ e dual, $\pi$.

Quando o arco $(k, l)$ é adicionado a $\mathcal{T}$, os arcos neste conjunto deixam de formar uma árvore uma vez que o arco $(k, l)$ formará um ciclo com outros arcos de $\mathcal{T}$. O passo seguinte consiste em retirar um dos arcos de $\mathcal{T}$, por forma a ficarmos novamente com uma árvore geradora. Obtém-se uma nova $EAG$ fazendo passar o máximo fluxo possível, $\Delta \geqslant 0$, através do ciclo $\mathcal{C}$, com o sentido de $(k, l)$ ou com sentido contrário, consoante o arco $(k, l)$ pertença a $L$ ou $U$, respectivamente, por forma a que a solução seguinte se mantenha admissível. O arco a retirar de $\mathcal{T}$, $(p, q)$, é um dos que não suporta um aumento do fluxo superior no ciclo.

Em resumo o algoritmo simplex para redes está descrito no Algoritmo 1.

*Algoritmo 1*: Algoritmo simplex para redes.
{ *Determina um fluxo de custo mínimo.* }

---

**input:** Rede orientada $\mathcal{N} = (\mathcal{G}, c, l, u, b)$.
**output:** Estrutura de árvore geradora óptima $EAG^{(k)}$.

```
(1)  begin
(2)    k ← 0;
(3)    Determinar uma EAG^(k) admissível inicial;
(4)    while (EAG^(k) não for óptima) do
(5)    begin
(6)      Determinar a EAG adjacente, EAG^(k+1):
(7)      begin
(8)        Considerar o terno (T, L, U), o fluxo x
(9)          e o vector das variáveis duais π associados com EAG^(k);
(10)       Determinar um arco de entrada, (k, l) e um arco de saída (p, q);
(11)       Adicionar (k, l) a T e retirar (p, q) de T;
(12)       Determinar a EAG e as soluções x e π associadas;
(13)     end
(14)   k ← k + 1, EAG^(k) ← EAG^(k+1);
(15)   end
(16) end
```

---

Duas *EAGs adjacentes* são obtidas uma a partir da outra, através da troca de um arco em $\mathcal{T}$ por um arco que não está em $\mathcal{T}$ através de uma operação de *pivotação* (eventualmente, o mesmo arco pode entrar e sair de $\mathcal{T}$, o que corresponde a mudar um arco do conjunto $L$ para o conjunto $U$ ou deste para o primeiro). As soluções associadas com estas $EAG$s são soluções básicas adjacentes. Consideremos o ciclo $\mathcal{C}$ que permite passar de uma $EAG$ para a sua adjacente enviando uma quantidade

$\Delta > 0$ de fluxo e as soluções associadas. Se enviarmos uma quantidade $\Delta_1$, tal que $0 < \Delta_1 < \Delta$, de fluxo através do mesmo ciclo obtemos uma solução diferente daquelas que é designada por *solução intermédia*.

Para obter uma solução admissível inicial, na linha (3) do Algoritmo 1, existem vários métodos entre os quais a resolução de um problema auxiliar através da denominada Fase I do método simplex ou, alternativamente, através da resolução de um problema de fluxo máximo na rede. A determinação de um fluxo admissível pode ser modelado como um problema de fluxo máximo, construindo uma rede incremental juntando um nodo origem $s$ e um nodo destino $t$ à rede $\mathcal{N}$. Para cada nodo $k$, com $b_k > 0$, adicionamos um arco $(s, k)$ ao conjunto $\mathcal{V}$, com capacidade $b_k$; para cada nodo $k$, com $b_k < 0$ adicionamos um arco $(k, t)$ a $\mathcal{A}$, com capacidade $-b_k$. Depois resolvemos o problema de fluxo máximo, na nova rede que designamos por $\mathcal{N}'$, do nodo $s$ para o nodo $t$. Pode mostrar-se que o problema tem um fluxo admissível se e só se no fluxo máximo os arcos com origem em $s$ estão saturados, isto é, se $x_{sj} = u_{sj}, \forall (s, j) \in \mathcal{A}'$, onde $\mathcal{A}'$ é o conjunto dos arcos da rede $\mathcal{N}'$.

Em cada iteração, o Algoritmo 1 gera sempre uma solução com componentes inteiras, para o problema FCM, devido às características da matriz das restrições que passa a ser uma matriz totalmente unimodular, isto é, uma matriz em que qualquer submatriz quadrada regular tem determinante igual a $+1$ ou $-1$ (ver Bazaraa *et al.*, 2005). Neste caso, a resolução do problema FICM pode ser feita relaxando a condição de que as variáveis são inteiras, obtendo-se sempre uma solução óptima inteira.

A utilização de duas $EAG$s adjacentes para determinar uma solução não-inteira é uma propriedade importante. Recordemos que, quando passamos de uma $EAG$ para uma adjacente, uma quantidade do fluxo $\Delta$ é enviada através de um ciclo $\mathcal{C}$. A quantidade $\Delta$ é inteira. Se enviarmos uma quantidade não-inteira, através do mesmo ciclo $\mathcal{C}$, é obtida uma solução não inteira. Esta solução tem exactamente $|\mathcal{C}|$ variáveis não inteiras e, como é evidente, não define uma $EAG$. Esta ideia é bastante importante, se pretendermos obter soluções não inteiras para o problema de restrição-$\varepsilon$ relaxado, que abordaremos mais adiante. Neste caso, a resolução do problema FICM é feita relaxando a restrição $x_{ij}$ inteiro, $\forall (i, j) \in \mathcal{A}$, porque existe pelo menos uma solução óptima com componentes inteiras, para o problema FCM que será óptima também para o problema FICM. Determinadas as soluções básicas óptimas (com componentes inteiras), as restantes podem ser determinadas através de combinações convexas destas (Bazaraa *et al.*, 2005).

**Proposição 2.6.1** *Consideremos o problema FCM com todas as componentes inteiras dos vectores c, u e b e duas EAGs adjacentes, $EAG'$ e $EAG''$ e o ciclo, $\mathcal{C}$, que permite passar de $EAG'$ para $EAG''$, quando uma quantidade do fluxo $\Delta$ é enviada através deste ciclo. Então, ao enviar uma quantidade não inteira através do ciclo $\mathcal{C}$, é obtida uma solução não inteira que contém exactamente $|\mathcal{C}|$ variáveis com valor não inteiro.*

*Demonstração:* O fluxo nos arcos $(i, j)$ em $EAG'$ é inteiro e na passagem de $EAG'$ para $EAG''$ apenas o fluxo nos arcos em $\mathcal{C}$ é alterado. A quantidade não inteira de fluxo enviada através do ciclo $\mathcal{C}$ é adicionada ou subtraída ao fluxo de cada arco no ciclo, consoante esse arco tem ou não o sentido do ciclo. Assim, o novo fluxo dos arcos no ciclo é não inteiro, enquanto nos outros arcos continua inteiro.

□

### 2.6.2 Algoritmo por eliminação dos circuitos de custo negativo

A optimalidade de uma solução, $x^*$ para o problema FCM pode ser avaliada através dos custos dos circuitos na rede residual. Pelo Teorema 2.5.3, uma solução admissível $x^*$ para o problema FCM é óptima se e só se a rede residual $\mathcal{N}(x^*)$ não contiver circuitos de custo negativo. Esta condição de optimalidade dá origem ao Algoritmo 2, para a determinação de um fluxo de custo mínimo do problema FCM. Este algoritmo, designado por eliminação dos circuitos de custo negativo, é baseado nas condições de optimalidade da inexistência de circuito negativo. Começa com um fluxo admissível e aumenta sucessivamente o fluxo através dos circuitos de custo negativo, na rede residual, até não existir qualquer ciclo de custo negativo.

A existência de um circuito negativo, $W$, na rede residual, pode ser detectado pela aplicação sucessiva do algoritmo do caminho mais curto modificado, entre cada par de nodos. Quando existe um circuito negativo, a distância entre dois nodos é infinitamente pequena e, deste modo, o algoritmo do caminho mais curto entra em ciclo, nunca conseguindo obter esse caminho, uma vez que em cada iteração o comprimento do caminho continua infinitamente a decrescer. No entanto, sabe-se que $-mM$ é um limite inferior de qualquer distância entre dois nodos, quando não existem na rede circuitos de custo negativo, onde $M$ é um número real positivo maior que o valor absoluto do custo de qualquer arco, isto é, $M \geqslant \max_{(i,j)\in\mathcal{A}}\{c_{ij}\}$. Assim, a obtenção de um comprimento inferior a este valor significa que existe um circuito de custo negativo. Esta informação usada em conjunto com o algoritmo do caminho mais curto permite detectar a existência de circuitos de custo negativo. Existem também outras formas de detectar um circuito de custo negativo (ver Ahuja *et al.*, 1993).

*Algoritmo 2*: Eliminação dos circuitos de custo negativo.
{*Determina um fluxo de custo mínimo.* }

---

**input:** Rede orientada $\mathcal{N} = (\mathcal{G}, c, l, u, b)$.
**output:** Solução de custo mínimo: $x$.

```
(1)  begin
(2)    Determinar um fluxo admissível x na rede;
(3)    while (N(x) contiver circuitos de custo negativo) do
(4)    begin
(5)      Identificar um circuito de custo negativo W;
(6)      δ ← min{r_ij : (i,j) ∈ W};
(7)      Adicionar δ unidades de fluxo ao ciclo W e restaurar N(x);
(8)    end
(9)  end
```

---

## 2.7 Problemas de fluxos em redes multi-objectivo

Nesta secção definimos os modelos de programação linear e programação linear inteira com vários objectivos. Fazemos uma revisão dos conceitos de eficiência e dominância, soluções não-dominadas e eficientes e sobre a conexidade do conjunto das soluções eficientes/não-dominadas.

### 2.7.1 Definição dos modelos de programação linear e programação linear inteira

O problema de fluxos em redes multi-objectivo (FRMO) obtém-se de forma semelhante ao problema FCM considerando agora mais de um objectivo. A nova rede associada difere da definida na Subsecção 2.1.2 para o problema FCM apenas nos custos associados a cada arco que, em vez de um único valor, são agora vectores de "custos", um "custo" para cada objectivo. A nova rede pode então ser escrita como $\mathcal{N} = (\mathcal{G}, (c^1, c^2, \cdots, c^p), u, b)$ onde $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, $u$ e $b$ estão definidos na Subsecção 2.1.2; $c^1, c^2, \cdots, c^p$ são vectores tais que para cada arco $(i,j) \in \mathcal{A}$, $c^1_{ij}, c^2_{ij}, \cdots, c^p_{ij}$ são os "custos" unitários do fluxo ao longo do arco $(i,j)$, relativos a cada um dos objectivos $1, 2, \cdots, p$.

O problema FRMO, associado à rede $\mathcal{N}$, pode ser formulado, matematicamente,

da seguinte forma:

$$\begin{array}{ll} \text{“minimizar”} & \Big( \sum\limits_{(i,j)\in\mathcal{A}} c_{ij}^1 x_{ij}, \sum\limits_{(i,j)\in\mathcal{A}} c_{ij}^2 x_{ij}, \cdots, \sum\limits_{(i,j)\in\mathcal{A}} c_{ij}^p x_{ij} \Big), \\ \text{sujeito a:} & \sum\limits_{(k,j)\in\mathcal{A}} x_{kj} - \sum\limits_{(i,k)\in\mathcal{A}} x_{ik} = b_k, \quad \forall k \in \mathcal{V}, \\ & 0 \leqslant x_{ij} \leqslant u_{ij}, \forall (i,j) \in \mathcal{A}, \end{array} \tag{2.13}$$

ou de forma mais compacta:

$$\begin{array}{ll} \text{“minimizar”} & f(x) = C^T x, \\ \text{sujeito a:} & x \in X, \end{array} \tag{2.14}$$

onde, $C$ é a matriz “custo” com colunas $c^1, c^2, \cdots, c^p$ e $c^k = (c_{i_1 j_1}^k, c_{i_2 j_2}^k, \cdots, c_{i_n j_n}^k)$, para $k = 1, 2, \cdots, p$.

A região admissível no espaço das variáveis de decisão, $X$, está definida em (2.3). A região admissível no espaço dos objectivos é um subconjunto de $\mathbb{R}^p$ assim definido:

$$\begin{array}{rl} Y = f(X) = \big\{ & y = (y_1, y_2, \cdots, y_p) \in \mathbb{R}^p : y_1 = \sum_{(i,j)\in\mathcal{A}} c_{ij}^1 x_{ij}, \\ & y_2 = \sum_{(i,j)\in\mathcal{A}} c_{ij}^2 x_{ij}, \cdots, y_p = \sum_{(i,j)\in\mathcal{A}} c_{ij}^p x_{ij}, \\ & x = (x_{i_1 j_1}, x_{i_2 j_2}, \cdots, x_{i_n j_n}) \in X \big\}. \end{array} \tag{2.15}$$

Se ao conjunto de restrições do problema FRMO juntarmos a condição de que o fluxo deve ser inteiro, isto é, $x_{ij}$ inteiro, $\forall (i,j) \in A$, obtemos o denominado problema de fluxos em redes inteiro multi-objectivo (FRIMO). A região admissível no espaço dos objectivos representa-se por $X^I$ e coincide com a mesma região no problema FICM (2.8). A região admissível no espaço dos objectivos é

$$\begin{array}{rl} Y^I = f(X^I) = \{ & y = (y_1, y_2, \cdots, y_p) \in \mathbb{R}^p : \\ & y^k = \sum\limits_{(i,j)\in\mathcal{A}} c_{ij}^k x_{ij}, k = 1, 2, \cdots, p, \\ & x = (x_{i_1 j_1}, x_{i_2 j_2}, \cdots, x_{i_n j_n}) \in X^I \}. \end{array} \tag{2.16}$$

Realcemos o caso em que a matriz “custo” tem todas as componentes inteiras o que implica que os vectores do conjunto $Y^I$ tenham também todas as suas componentes inteiras, uma vez que por hipótese o fluxo também tem todas as suas componentes inteiras.

Como casos particulares dos problemas FRMO e FRIMO, designamos por FRBO, FRIBO os problemas de fluxos em redes contínuo e inteiro bi-objectivo, respectivamente.

### 2.7.2 Definição dos diferentes tipos de soluções eficientes/não-dominadas

Consideremos um problema FRMO. Neste problema pretendemos minimizar em simultâneo um conjunto de $p$ funções objectivo lineares. Em geral, não existe qualquer solução que minimize em simultâneo todos os objectivos, dizemos, por isso, que os objectivos estão em conflito. Deste modo, deveremos definir o que se entende por resolução de um problema multi-objectivo, isto é, qual o significado do termo "minimizar" num problema de programação linear com mais de um objectivo. A primeira referência conhecida, de como tratar esta questão, é atribuída a Pareto (1896) e será a utilizada aqui também. Na resolução do problema, utilizando o conceito de optimalidade de Pareto, são procuradas as soluções eficientes. As definições seguintes são baseadas em Steuer (1986).

**Definição 2.7.1 (Dominância)** *Consideremos duas soluções $y'$ e $y''$ em $Y$. Dizemos que $y'$ domina $y''$ se $y' \leq y''$ , isto é, $y'_q \leqslant y''_q$, para todo $q = 1, \ldots, p$, com pelo menos uma desigualdade estrita.*

**Definição 2.7.2 (Solução não-dominada)** *Uma solução $y' \in Y$ diz-se não-dominada (ND) se não existir outra solução $y \in Y$ tal que $y \leq y'$. Caso contrário, $y'$ diz-se uma solução dominada. O conjunto de todas as soluções não-dominadas em $Y$ será designado por $ND(Y)$.*

**Definição 2.7.3 (Solução eficiente)** *Uma solução $x' \in X$ diz-se eficiente ou óptima de Pareto se não existir outra solução $x \in X$ tal que $y = f(x) \leq y' = f(x')$. O conjunto de todas as soluções eficientes será designado por $EF(X)$.*

Podemos ainda dizer, de forma equivalente, que uma solução $x' \in X$ é eficiente se não existir outra solução $x \in X$ com um melhor valor para um dado objectivo sem deteriorar o valor de pelo menos um dos restantes objectivos, ou ainda, se não existe uma solução $y \in Y$ que domine $y' = f(x')$.

As definições anteriores podem ser também utilizadas para o problema FRIMO. Neste problema, tanto a região admissível, $X^I$, no espaço das variáveis de decisão, como a região admissível, $Y^I$, no espaço dos objectivos (admitindo todas as componentes da matriz $C$ inteiras), são conjuntos cujos elementos são vectores com componentes inteiras. Enquanto no problema FRMO, os conjuntos $EF(X)$ e $ND(Y)$ são subconjuntos da fronteira de $X$ e $Y$, respectivamente, os conjuntos das soluções eficientes e não-dominadas, no problema FRIMO, não pertencem necessariamente às fronteiras das envolventes convexas dos conjuntos $X^I$ e $Y^I$, $conv(X^I)$ e $conv(Y^I)$, respectivamente. Podemos desta forma ter soluções eficientes na fronteira ou no interior de $conv(X^I)$ ou soluções não-dominadas na fronteira ou no interior de $conv(Y^I)$.

Designemos por $F$ uma face de $Y$. $F$ diz-se *não-dominada* se todos os pontos de $F$ são não-dominados, isto é, se para cada um dos pontos a solução, cujas componentes são as coordenadas desse ponto é não-dominada. Uma face $F \subseteq Y$ de dimensão $r$ diz-se *não-dominada máxima* se não existir qualquer face $F' \subseteq Y$ não-dominada com dimensão $r'$ tal que $F \subset F'$ e $r < r'$. A imagem inversa, em $X$, de uma face não-dominada designa-se por *face eficiente* e de uma face não-dominada máxima por *face eficiente máxima*.

Designemos por

$$Y^{\geqq} = Conv(Y^I + \mathbb{R}^p_{\geqq}),$$

onde $\mathbb{R}^p_{\geqq} = \{y \in \mathbb{R}^p | y \geqq 0\}$ e $Y^I + \mathbb{R}^p_{\geqq} = \{y \in \mathbb{R}^p : y = y' + y'', y' \in Y^I \text{ e } y'' \in \mathbb{R}^p_{\geqq}\}$, $y \geqq 0$ se $y_q \geqslant 0$, $q = 1, 2 \cdots, p$.

**Definição 2.7.4 (Solução suportada e não-suportada)** *Designemos por y uma solução não-dominada para o problema FRIMO. Então, se y pertence à fronteira do conjunto $Y^{\geqq}$ diz-se uma solução não-dominada suportada. Caso contrário, y diz-se uma solução não-dominada não-suportada.*

**Definição 2.7.5 (Solução extrema ND )** *Dizemos que uma solução não-dominada y é uma solução extrema se é um ponto extremo do poliedro representado por $Y$.*

Imagens inversas de soluções não-dominadas suportadas dizem-se *soluções eficientes suportadas* e imagens inversas de soluções não-dominadas não-suportadas dizem-se *soluções eficientes não-suportadas*.

### 2.7.3 Conexidade do conjunto das soluções não-dominadas

A conexidade de um conjunto é uma propriedade importante na determinação dos elementos desse conjunto, uma vez que assegura que não existem pontos isolados. O conhecimento de como estes elementos estão ligados permite a elaboração de algoritmos para os determinar. No nosso caso estamos interessados em saber se os conjuntos das soluções eficientes/não-dominadas para os problemas FRMO e FRIMO, referidos acima, são ou não conexos. Durante os últimos anos foram feitos estudos a este respeito. Naccache (1978) mostrou que para qualquer problema linear multi-objectivo o conjunto das soluções não-dominadas, $ND(Y)$, é um conjunto conexo. Em particular, o problema FRMO é um caso particular do problema de programação linear multi-objectivo e, como tal, é conexo. Podemos também concluir, a partir do trabalho realizado por Warburton (1983), que o conjunto das soluções eficientes

para o problema FRMO é conexo. O trabalho de Isermann (1977) leva-nos a concluir que o conjunto das $EAG$s eficientes é conexo, no sentido em que é conexo o grafo cujo conjunto de nodos é formado pelas $EAG$s eficientes e em que existe uma aresta entre dois nodos se e só se as $EAG$s correspondentes podem ser obtidas uma a partir da outra através de uma operação de *pivotação*.

No caso do problema FRIMO, cujo conjunto de soluções não-dominadas é formado pelo conjunto das soluções suportadas e não-suportadas, apenas conseguimos encontrar o trabalho de Ehrgott (1999), onde se afirma que o conjunto das soluções suportadas define um grafo conexo. A demonstração deste teorema é feita assumindo que os pontos inteiros de uma face não-dominada estão num grafo conexo cujos nodos são os pontos associados com as $EAG$s dessa face e os pontos nos segmentos de recta entre cada duas $EAG$s adjacentes, que permitem a ligação dos nodos. Como veremos, a conexidade das soluções suportadas não pode ser provada desta forma. Mostraremos neste trabalho que o conjunto de todas as soluções suportadas pode ser identificado de forma diferente e a conexidade definida também de forma diferente.

# Capítulo 3

# Estado-da-arte sobre problemas de fluxos em redes multi-objectivo

Neste capítulo fazemos uma revisão da literatura sobre o problema de fluxos em redes multi-objectivo. O capítulo está dividido em quatro secções. Na Secção 3.1 é feita uma introdução ao capítulo. Na Secção 3.2 são revistos os principais artigos dedicados a algoritmos exactos. Esta secção está ainda dividida em duas subsecções de acordo com o tipo de variáveis consideradas, contínuas, onde são revistos os artigos Lee & Pulat (1991), Malhotra & Puri (1984), Pulat *et al.* (1992), Sedeño Noda & González-Martín (2000) e Sedeño Noda & González-Martín (2003) e inteiras, onde são revistos os artigos Figueira (2000, 2001), Gouveia (2002), Lee & Pulat (1993), Sedeño Noda & González-Martín (2001) e Raith & Ehrgott (2009). Na Secção 3.3 são revistos os artigos sobre métodos aproximados e, tal como a anterior, é dividida em duas subsecções de acordo com o tipo de variáveis consideradas, contínuas e discretas. Os artigos revistos nesta secção são: Hamacher *et al.* (2007a), Mustafa & Goh (1998), Nikolova (1998, 2001) e Nikolova (2003). Na Secção 3.4 é feita uma análise crítica dos métodos exactos, revendo os erros cometidos que impedem que os algoritmos propostos calculem todas as soluções eficientes/não-dominadas, como referido nos respectivos artigos.

## 3.1 Introdução

Os problemas de fluxos em redes são casos particulares do problema de programação linear e, como tal, podem ser resolvidos utilizando os algoritmos para este. Sabemos, no entanto, que, para problemas com um único objectivo, vários algoritmos foram elaborados, utilizando as propriedades particulares dos problemas de fluxos em redes. Estes algoritmos são computacionalmente mais eficientes, no sentido em que utilizam menos recursos computacionais, para resolver o mesmo problema, como, por exemplo, memória e tempo. São exemplos o algoritmo Húngaro para problemas de afectação, o algoritmo de Dijkstra para problemas do caminho mais curto e o algoritmo simplex para problemas do fluxo de custo mínimo numa rede. Motivados por estes resultados, também no caso de problemas com vários objectivos se concebem algoritmos mais eficientes que possam beneficiar das propriedade particulares destes. Apesar de encontrarmos alguns trabalhos nesta área, podemos dizer que ainda há um grande caminho a percorrer. A maior parte dos artigos encontrados, para este tipo de problemas, apenas tratam o problema bi-objectivo. A investigação no tema aqui abordado parece-nos plenamente justificada pela relevância da solução deste tipo de problemas na tomada de decisão em problemas práticos e, também, pela quantidade de questões teóricas a ele associadas.

## 3.2 Algoritmos exactos

Consideramos como pertencendo à classe dos algoritmos exactos todos os algoritmos que dizem determinar o conjunto de todas as soluções eficientes ou não-dominadas para o problema considerado. Dividimos esta classe de algoritmos em duas: caso contínuo e caso inteiro, correspondendo assim aos algoritmos para os problemas FRMO e FRIMO, respectivamente. Nos problemas FRMO, as soluções eficientes estão na fronteira da envolvente convexa da região admissível, enquanto, no problema FRIMO, podem existir soluções eficientes no interior dessa região, o que torna o cálculo de todas estas soluções mais difícil.

### 3.2.1 Caso contínuo

Nesta subsecção fazemos uma revisão sobre os algoritmos existentes que pretendem calcular o conjunto de todas as soluções eficientes ou todas as soluções não-dominadas para o problema FRMO. A maioria dos artigos, no caso contínuo, exploram a conexidade do conjunto das soluções eficientes. Convém recordar, em primeiro lugar, que o problema FRMO é um caso particular do problema de programação linear multi-objectivo e, como tal, os métodos para o segundo problema são também

aplicáveis ao primeiro. Assim, por exemplo, o problema FRMO pode ser resolvido utilizando o *software* ADBASE baseado no algoritmo para problemas de programação linear multi-objectivo em Steuer (1986) e revisto em Schechter & Steuer (2005). Também o estudo feito por Isermann (1977) sobre a determinação do conjunto de todas as soluções eficientes para o problema de programação linear multi-objectivo se pode aplicar ao problema FRMO. Neste estudo é mostrado que é conexo o grafo $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, onde $\mathcal{V}$ representa o conjunto das soluções básicas eficientes e $\mathcal{A}$ o conjunto de arestas $\{i, j\}$ tais que as soluções básicas eficientes representadas pelos nodos $i$ e $j$ podem ser obtidas uma a partir da outra por uma simples operação de *pivotação*. Esta propriedade permite-nos começar arbitrariamente numa solução básica eficiente e, a partir desta, obter o conjunto de todas as soluções básicas eficientes restantes, através da realização de operações de *pivotação*. A grande dificuldade da procura de soluções eficientes desta forma surge quando as soluções encontradas são degeneradas. Segundo Cunningham (1976) mais de 90% dos *pivots* no método simplex para redes podem ser degenerados. No entanto, é conhecido o bom comportamento do método simplex para redes (ver Bazaraa *et al.*, 2005). No problema FCM são conhecidas as vantagens de utilizarmos $EAG$s em vez da notação matricial dos problemas de programação linear (Bazaraa *et al.*, 2005).

A revisão que faremos aqui refere-se a algoritmos específicos para o problema FRMO, utilizando, por isso, propriedades especiais destes, com o objectivo de minimizar o tempo necessário para resolver o problema assim como a quantidade de memória envolvida. Os artigos revistos nesta subsecção são os seguintes: Lee & Pulat (1991), Malhotra & Puri (1984), Pulat *et al.* (1992) e Sedeño Noda & González-Martín (2000, 2003).

**Malhotra & Puri (1984)** Malhotra & Puri (1984) apresentaram um algoritmo que, segundo eles, calcula todas as soluções não-dominadas do problema FRBO, quando as capacidades dos arcos são todas iguais, isto é, $u_{ij} = u, \forall (i, j) \in \mathcal{A}$.

O algoritmo começa por calcular o mínimo lexicográfico para o primeiro objectivo (a solução que minimiza o primeiro objectivo ou, se existir mais que uma, de entre as várias soluções que minimizam o primeiro objectivo aquela que tem o menor valor para o segundo objectivo, ou uma delas se existir mais que uma), utilizando o algoritmo dos arcos não-conformes (*out-of-kilter*). A solução obtida é óptima para o primeiro objectivo e, por isso, todos os arcos estão em conformidade relativamente a este objectivo. Admitindo que os mínimos para os dois objectivos são diferentes, existe pelo menos um arco não-conforme para o segundo objectivo. Estes arcos podem ser de dois tipos, consoante o fluxo no arco e o custo reduzido, são ambos positivos ou o fluxo tem valor inferior à capacidade e o custo reduzido é negativo. Para cada um destes arcos são procurados os caminhos incrementais, a partir dos quais uma nova

solução é gerada, continuando assim o processo até o arco se tornar conforme (*in--kilter*), para o segundo objectivo. O conjunto das soluções não-dominadas extremas é obtido a partir destas, por um processo de comparação. Determinadas as soluções extremas, as restantes obtêm-se por combinação convexa de pares adjacentes.

Verificou-se que este algoritmo estava incorrecto. No exemplo apresentado no seu artigo, Malhotra e Puri terminam com um exemplo cujo conjunto de soluções não-dominadas extremas obtido é $\{(60, 288), (96, 120), (156, 60)\}$. É fácil verificar que falta neste conjunto a solução $(72, 204)$.

**Lee & Pulat (1991)** Lee & Pulat (1991) propuseram um novo algoritmo, para a determinação do conjunto das soluções não-dominadas do problema FRBO. Estes autores recorrem à determinação das soluções óptimas de um problema paramétrico. Como pode ser visto no artigo de Geoffrion (1967), as soluções óptimas do problema paramétrico (3.1) para $\lambda \in ]0, 1[$ são soluções eficientes para o problema FRBO e estas são soluções óptimas do problema (3.1) para algum $\lambda$ no intervalo $]0, 1[$.

O algoritmo começa por determinar uma solução eficiente inicial e a solução não--dominada correspondente. Esta solução inicial é uma solução óptima do problema paramétrico

$$\begin{array}{ll} \text{minimizar} & (\lambda c^1 + (1-\lambda)c^2)x, \\ \text{sujeito a:} & x \in X, \end{array} \tag{3.1}$$

com $\lambda = 1$. A resolução deste problema é feita utilizando o algoritmo dos arcos não-conformes. No final, a solução obtida, se não for básica, é reajustada por forma a que a primeira solução considerada seja básica. Depois procura na fronteira da região admissível o conjunto das soluções não-dominadas: para a esquerda, considerando os arcos não-conformes relativamente ao primeiro objectivo; para a direita, considerando os arcos não-conformes relativamente ao segundo objectivo. Para mudar de uma solução para a adjacente, faz entrar na base um arco com o menor quociente entre os custos reduzidos para os dois objectivos. O valor absoluto deste quociente é igual ao declive da recta que contém o ponto inicial e o ponto associado à solução adjacente a calcular.

Na elaboração deste algoritmo, os autores desprezaram os casos em que existem soluções degeneradas. Além disso, supõem, de forma errada, que qualquer solução eficiente extrema no espaço das variáveis de decisão tem como imagem uma solução não-dominada extrema no espaço dos objectivos.

**Pulat *et al.* (1992)** Pulat *et al.* (1992) propuseram um novo algoritmo para calcular o conjunto de todas as soluções não-dominadas do problema FRBO.

Tal como em Lee & Pulat (1991), também Pulat *et al.* (1992) recorrem à resolução

do problema paramétrico. O algoritmo proposto começa por determinar o mínimo lexicográfico, para o primeiro objectivo, da seguinte forma:

a) calcular o mínimo para o primeiro objectivo na região admissível do problema FRBO, usando o método simplex para redes;

b) minimizar o valor do segundo objectivo, na mesma região admissível, sem deteriorar (aumentar) o valor do primeiro objectivo.

A solução obtida é básica e a sua imagem é o ponto extremo mais à esquerda da região admissível, no espaço dos objectivos. Esta solução é uma solução óptima do problema paramétrico:

$$\begin{array}{ll} \text{minimizar} & f(x) = (c^1 + \lambda c^2)x, \\ \text{sujeito a:} & x \in X, \end{array}$$

com $\lambda = 0$. De seguida, o algoritmo procura as soluções óptimas deste problema com $0 \leqslant \lambda \leqslant \phi$, sendo $\phi$ um número finito algebricamente grande, passando de uma solução para as não-dominadas adjacentes. O ponto extremo actual e cada um dos pontos extremos adjacentes formam um segmento de recta. Pertencem à fronteira não-dominada os pontos nos segmentos que fazem parte das rectas com maior declive. O declive é determinado pelo quociente dos custos reduzidos associados aos arcos não-básicos da solução actual. O processo é repetido para cada novo ponto extremo encontrado até todos terem sido analisados.

Pulat *et al.* (1992) perceberam que a imagem de pontos extremos da região admissível no espaço das variáveis de decisão nem sempre é um ponto extremo da região admissível no espaço dos objectivos. Consideraram todas as soluções óptimas alternativas do problema paramétrico e, deste modo, têm a garantia que encontram todos os pontos extremos, no espaço dos objectivos. No final, o conjunto de todas as soluções não-dominadas pode ser obtido, determinando combinações convexas das soluções não-dominadas extremas adjacentes.

**Sedeño Noda & González-Martín (2000)** Sedeño Noda & González-Martín (2000) propuseram um novo algoritmo para calcular o conjunto de todas as soluções extremas da região admissível, no espaço dos objectivos, que são não-dominadas do problema FRBO.

O algoritmo proposto é baseado no algoritmo de Lee & Pulat (1991). A diferença, segundo eles, reside no facto deste algoritmo calcular soluções não-dominadas que não são pontos extremos, enquanto o destes autores apenas calcula soluções não--dominadas extremas. Além disso, Sedeño Noda & González-Martín (2000) usam

agora o método simplex para redes, em vez do método dos arcos não-conformes, para a resolução do problema paramétrico.

O algoritmo começa com o ponto extremo $f(x^0)$ no espaço dos objectivos, correspondente à solução não-dominada que resulta da resolução do problema paramétrico (3.1) com $\lambda = 1$. Depois, determina os restantes pontos extremos através de uma sequência finita de operações de *pivotação*. $f(x^0)$ é o ponto extremo mais à esquerda, no espaço dos objectivos, o algoritmo procede da esquerda para a direita encontrando os restantes pontos extremos até obter o correspondente ao mínimo do segundo objectivo. Para isso, é considerada a solução associada ao ponto extremo actual, inicialmente $x^0$, que corresponde a uma $EAG$ $(\mathcal{T}, L, U)$. Esta solução é óptima para (3.1) para pelo menos um valor de $\lambda$ no intervalo $]0, 1[$. De seguida, é determinado o conjunto de arcos associados ao menor quociente entre os custos reduzidos do segundo e primeiro objectivos, dos arcos em $L$ e em $U$ com custo reduzido para o segundo objectivo negativo e positivo, respectivamente, isto é, entre os arcos que não preenchem as condições de optimalidade para o segundo objectivo. Segundo eles, entre este conjunto de arcos está o *pivot* que conduz ao ponto não-dominado extremo adjacente, no espaço dos objectivos.

Apesar deste algoritmo calcular todos os pontos não-dominados extremos no espaço dos objectivos, não calcula apenas estes pontos como é dito pelos autores, isto é, no conjunto resultante da aplicação do algoritmo podem existir outras soluções não-dominadas para além das extremas (ver Raith & Ehrgott, 2009).

**Sedeño Noda & González-Martín (2003)** Sedeño Noda & González-Martín (2003) propuseram um método para determinar todos os pontos não-dominados extremos (no espaço dos objectivos), que é uma modificação do método proposto por Aneja & Nair (1979) para o problema de transportes. Este método, contrariamente ao que acontecia com os anteriores, não utiliza a conexidade do conjunto das soluções eficientes. Recorre ao uso da teoria Lagrangeana para criar um problema auxiliar, para a determinação das soluções não-dominadas. Segundo os autores, os métodos baseados na exploração da adjacência devem calcular soluções degeneradas de soluções básicas eficientes o que se traduz num custo computacional muito alto, uma vez que, segundo Cunningham (1976), mais de 90% dos *pivots*, no método simplex para redes, podem ser degenerados. De qualquer modo chegam à conclusão que o método proposto é mais lento que o de Sedeño Noda & González-Martín (2000), que usa o conceito de adjacência. Argumentam, no entanto, que este método tem vantagens no sentido em que pode ser um guia na procura de soluções eficientes tendo em conta as preferências do decisor e, além disso, usam o método quando estão apenas interessados num subconjunto de pontos não-dominados extremos, numa região particular do espaço dos objectivos.

O método começa por determinar o mínimo lexicográfico, para cada uma das funções. Para cada dois pontos identificados consecutivamente procura pontos extremos adicionais no triângulo entre estes pontos e um ponto ideal local. Procura o ponto extremo mais próximo da linha recta que passa pelo ponto ideal e um ponto, designado por *nível de aspiração*, com componentes superiores à do *ponto ideal*. Este novo ponto encontrado define duas novas áreas de pesquisa futura.

### 3.2.2 Caso inteiro

Com excepção do artigo Ehrgott (1999), que propõe um método para resolver o problema FRIMO, todos os restantes encontrados tratam o caso de problemas com dois objectivos e não mais. O problema de encontrar todas as soluções eficientes ou não-dominadas é quase sempre dividido em duas fases: na primeira fase procuram-se todas as soluções inteiras eficientes suportadas e, numa segunda fase, procuram-se as soluções inteiras eficientes não-suportadas. Quase todos os artigos que propõem algoritmos, para a determinação de todas as soluções não-dominadas para o problema FRIBO, utilizam na Fase I o método proposto por Lee & Pulat (1993). Os artigos revistos nesta subsecção são: Ehrgott (1999), Figueira (2000), Gouveia (2002), Lee & Pulat (1993), Raith & Ehrgott (2009) e Sedeño Noda & González-Martín (2001).

**Lee & Pulat (1993)** Lee & Pulat (1993) propõem um algoritmo para o cálculo de todas as soluções não-dominadas para o problema FRIBO dividido em duas fases. Na primeira fase, o algoritmo procura o conjunto das soluções não-dominadas suportadas do problema e, na segunda fase, o conjunto das soluções não-dominadas não-suportadas.

Na Fase I, Lee & Pulat (1993) remetem para o artigo Lee & Pulat (1991) onde, na Secção 5, página 126, descrevem o cálculo destas soluções da seguinte forma: Consideram dois pontos extremos $x^i$ e $x^{i+1}$, correspondentes a soluções eficientes adjacentes (calculadas utilizando o algoritmo proposto) e designam por $y^i = (y_1^i, y_2^i)$ e $y^{i+1} = (y_1^{i+1}, y_2^{i+1})$ as respectivas imagens, no espaço dos objectivos. Como todos os pontos extremos do problema FRBO são inteiros, sugerem a seguinte forma de calcular todas as soluções inteiras nas arestas não-dominadas: no final dos Passos 2 e 3, no momento em que a divisão é feita, guardar a etiqueta final de $i^*$, designada por $q^{i^*}$. Então todas as soluções inteiras na aresta eficiente ligando $y^i$ e $y^{i+1}$ seriam obtidas da seguinte forma:

$$I' = y^i + \alpha \frac{y^{i+1} - y^i}{q^{i^*}},$$

quando $\alpha = 1, 2, \cdots, q^{i^*} - 1$. Esta forma de proceder é recordada por Hamacher *et al.* (2007b), página 1413.

Desconhecendo o que se passa nos Passos 2 e 3, importa realçar aqui que o que é proposto é que as soluções não-dominadas inteiras no segmento que liga duas soluções não-dominadas extremas, $y^i$ e $y^{i+1}$, sejam calculadas como uma combinação linear destas. Este procedimento nem sempre determina todas as soluções suportadas como veremos no Capítulo 6 e como pode ser visto também em Eusébio & Figueira (2009b).

Na Fase II do algoritmo procuram-se as soluções não-dominadas não-suportadas. Considera-se um par de soluções adjacentes, $x^t$ e $x^{t+1}$, onde $x^{t+1}$ foi obtida de $x^t$ fazendo entrar o arco $(u, v)$ na base. Para cada arco $(i, j) \neq (u, v)$ não-básico, destas soluções, cujo fluxo $x_{ij}$ é igual ao seu limite inferior $l_{ij}$ em ambas as soluções $x^t$ e $x^{t+1}$, determina-se o conjunto das soluções suportadas do problema FRBO modificado através da alteração do limite inferior de fluxo no arco $(i, j)$ para $l_{ij} + 1$, $l_{ij} + 2$, $\cdots$, $u_{ij}$. O mesmo raciocínio é utilizado para arcos não-básicos, $(i, j)$, diferentes de $(u, v)$ com fluxo igual à sua capacidade, $x_{ij} = u_{ij}$, agora com a capacidade do fluxo máximo modificada para $u_{ij} - 1, u_{ij} - 2, \cdots, l_{ij}$. As soluções encontradas são depois analisadas para ver se podem ser soluções não-dominadas do problema inicial. São deduzidas algumas propriedades e enunciadas algumas regras que evitam que sejam calculadas todas as soluções para todos os problemas auxiliares.

Neste algoritmo não são considerados problemas com soluções degeneradas, o que é bem frequente neste tipo de problema. A generalização deste algoritmo a casos deste tipo é feito em Huarng *et al.* (1992).

Sedeño Noda & González-Martín (2001) afirmam que este algoritmo não determina o conjunto de todas as soluções não-dominadas do problema FRIBO, argumentando que a razão se deve ao facto de apenas serem introduzidos dois arcos não-básicos $(i, j)$ e $(u, v)$ de cada vez, quando mais do que dois são necessários, em geral. Então propuseram um novo algoritmo para este problema.

**Ehrgott (1999)** Ehrgott (1999) propõe um método para determinar o conjunto das soluções não-dominadas do problema FRIMO em duas fases: na primeira fase é determinado o conjunto de todas as faces não-dominadas máximas para o problema relaxado, isto é, sem exigir que o fluxo nos arcos seja inteiro; na segunda fase determina o conjunto das soluções não-dominadas. São enumerados como passos gerais os seguintes:

1. Determinar uma solução não-dominada extrema $y^1$.
2. Partindo de $y^1$ determinar todas as soluções não-dominadas extremas.
3. Determinar todas as faces máximas não-dominadas que formam a fronteira não-dominada.

4. Determinar as soluções inteiras na fronteira não-dominada.

5. Determinar todas as soluções não-dominadas no "interior" da região admissível no espaço dos objectivos.

A determinação de uma solução extrema inicial é feita resolvendo o problema

$$\begin{aligned} \text{minimizar} \quad & \sum_{q=1}^{p} \sum_{(i,j)\in A} \lambda_q c_{ij}^q x_{ij}, \\ \text{sujeito a:} \quad & x \in X, \end{aligned}$$

usando o algoritmo dos arcos conformes. Se a solução óptima encontrada pelo algoritmo para este problema não for básica, procede-se a uma alteração de fluxo de forma a encontrar uma outra solução básica com o mesmo custo. O conjunto de soluções não-dominadas extremas é determinado partindo da solução anterior e utilizando o conceito de adjacência, verificando para todos os pontos extremos adjacentes dos já determinados, se podem ser eficientes extremos. As faces máximas não-dominadas são determinadas verificando todas as faces associadas com o grafo em que os nodos representam o conjunto das soluções básicas eficientes e em que uma aresta entre dois nodos existe se e só se estes nodos correspondem a soluções básicas adjacentes, como em Isermann (1977). A determinação dos pontos inteiros nas faces máximas não-dominadas é feita da seguinte forma: a) durante a determinação das soluções não-dominadas extremas todos os que pertencem às arestas não-dominadas podem ser determinados imediatamente; b) considera-se uma face máxima não-dominada $F'$ de dimensão $t$, $2 \leqslant t \leqslant p-1$ e um ponto extremo, $y^1$ de $F'$. Escolhe-se de seguida $t$ soluções não-dominadas extremas adjacentes $y^2, \cdots, y^{t+1}$ de tal forma que $\{y^1, \cdots, y^{t+1}\}$ sejam independentes. Isto identifica variáveis não-básicas $x_{uv_i}$ em $y^1$. Então todas as combinações inteiras de mudança no fluxo das variáveis $x_{uv_i}$ (e os ciclos respectivos) conduzindo de $y^1$ às soluções extremas adjacentes $y^2, \cdots, y^{t+1}$ definem o conjunto das soluções inteiras não-dominadas em $F'$. As mudanças unitárias de fluxo a serem consideradas estão entre 1 e a alteração de fluxo necessária para atingir $y^i$, respectivamente. O conjunto das restantes soluções não-dominadas, isto é, das não-suportadas é determinado pelo paralelismo de faces em novos problemas criados do inicial modificando os limites dos arcos conformes para todos os objectivos não sendo uma solução eficiente.

Como se percebe e é dito pelo autor (linha 9, página 234) o procedimento para determinar o conjunto de soluções não-dominadas suportadas é a generalização para mais de dois objectivos do que foi feito por Lee & Pulat (1993) para dois objectivos. Sendo assim, podemos afirmar que nem sempre o conjunto de todas as soluções não-dominadas é determinado.

**Figueira (2000)** Figueira (2000) propõe um algoritmo para a determinação do conjunto de todas as soluções não-dominadas para o problema FRIBO. O algoritmo proposto está dividido em três partes da seguinte forma:

1. Na primeira parte determina o conjunto de todas as soluções não-dominadas suportadas que são pontos extremos da região admissível, no espaço dos objectivos. Para isso, calcula o mínimo lexicográfico para o primeiro objectivo e, partindo deste ponto, calcula os restantes pontos extremos por ordem decrescente do valor do segundo objectivo e por ordem crescente do valor do primeiro objectivo. Considera a solução actual e a $EAG = (\mathcal{T}, L, U)$ associada e passa para a solução adjacente na fronteira suportada da região admissível, do espaço dos objectivos, com um valor superior ou igual para o primeiro objectivo. Para isso, utiliza o valor dos custos reduzidos para identificar o arco que deve entrar em $\mathcal{T}$ de tal modo que uma *pivotação* leve à solução pretendida. Este caminho entre os pontos extremos existe como pode ser confirmado pelo trabalho de Isermann (1977).

2. Na segunda parte procuram-se algumas soluções suportadas intermédias, para reduzir o espaço de pesquisa, recorrendo ao método proposto por Lee & Pulat (1991).

3. Finalmente, é determinado o conjunto das soluções não-dominadas, restantes. Verifica-se que cada solução não-dominada do problema FRIBO é uma solução óptima do problema de restrição-$\varepsilon$

$$\begin{array}{ll} \text{minimizar} & c^1 x, \\ \text{sujeito a:} & x \in X, \\ & c^2 x \leqslant \varepsilon, \end{array} \tag{3.2}$$

   para uma escolha adequada do valor $\varepsilon$. A resolução do problema (3.2) é feita utilizando o procedimento de separação e avaliação progressivas (PSAP), permitindo trabalhar sempre com o problema de fluxos em redes em vez da resolução de um problema de programação linear mais geral.

O algoritmo proposto aqui determina o conjunto de todas as soluções não-dominadas, apesar de, no segundo passo, não determinar todas as suportadas. O passo 2 pode mesmo ser abandonado (ver Eusébio & Figueira, 2009a). Este algoritmo também pode trabalhar com o fenómeno da degenerescência, referido aqui como uma maior dificuldade.

**Sedeño Noda & González-Martín (2001)** Sedeño Noda & González-Martín (2001) propõem um algoritmo para o cálculo de todas as soluções não-dominadas do problema FRIBO.

O algoritmo é dividido em duas fases. Na Fase I são calculadas as soluções não-dominadas suportadas. A estratégia utilizada é semelhante à de Lee & Pulat (1993). O algoritmo determina um ponto extremo no espaço dos objectivos, em seguida os pontos não-dominados extremos (no espaço dos objectivos) e também os pontos inteiros não-dominados entre ambos. Para gerar os pontos inteiros não-dominados entre dois pontos extremos adjacentes $f(x^i)$ e $f(x^j)$ eles consideram uma variável não-básica $x_{uv}$ do conjunto $S$ de todas as variáveis não-básicas $x_{uv}$ tais que

$$\frac{\bar{c}^2_{uv}}{\bar{c}^1_{uv}} = \min\left\{\frac{\bar{c}^2_{ij}}{\bar{c}^1_{ij}} : \bar{c}^2_{ij} < 0, \forall (i,j) \in L \text{ ou } \bar{c}^2_{ij} > 0, \forall (i,j) \in U\right\}.$$

Designemos por $\delta_{uv}$ a alteração de fluxo da variável $x_{uv}$, quando esta entra na base na iteração do método simplex que origina a solução $x^j$ a partir de $x^i$ e por $\mathcal{C}$ o conjunto de arcos cujo fluxo foi alterado. Adicionando ou subtraindo, de acordo com o sentido do arco em $\mathcal{C}$, uma unidade de fluxo aos arcos de $\mathcal{C}$ obtemos mais $\delta_{uv} - 1$ soluções que estão no segmento de recta que une os pontos $x^i$ e $x^j$ e as suas imagens no segmento de recta que une os pontos $f(x^i)$ e $f(x^j)$. Desta forma dizem determinar todas as soluções não-dominadas suportadas. Este método para determinar as soluções suportadas nem sempre determina o conjunto de todas estas soluções, como já referimos anteriormente.

Na Fase II calculam as soluções não-dominadas não-suportadas. Para isso, começam por utilizar as soluções calculadas na Fase I para reduzir a área de procura destas soluções a um conjunto de triângulos-rectângulos cujas hipotenusas são os pontos correspondentes a duas soluções não-dominadas suportadas consecutivas. Dentro de cada triângulo as soluções não-dominadas são calculadas adoptando a mesma estratégia de Lee & Pulat (1993), mas agora, para além de considerar todas os arcos não-básicos, em cada solução não-dominada suportada consideram também perturbações em mais do que um arco não-básico ao mesmo tempo. A cada $EAG$, $(\mathcal{T}^k, L^k, U^k)$, obtida na Fase I é associada uma lista, $R$, de arcos não-básicos para os quais uma alteração de fluxo no ciclo que leva à $EAG$ adjacente pode conduzir a uma solução não-dominada. O conjunto $R$ é definido da seguinte forma:

$$R = \{(i,j) \in L^k \text{ e } (i,j) \in U^k : \ \hat{c}^1_{ij} > 0, \hat{c}^2_{ij} < 0 \text{ e } s_{ij} > s_{uv}\},$$

onde $\hat{c}_{ij} = \bar{c}_{ij}$ se $(i,j) \in L^k$ ou $\hat{c}_{ij} = -\bar{c}_{ij}$ se $(i,j) \in U^k$, $s_{ij} = \hat{c}^2_{ij}/\hat{c}^1_{ij}$ e $s_{uv}$ é o declive do segmento de recta entre o ponto representado pela $EAG$ actual e o ponto representado pela solução adjacente, obtida quando $(u,v)$ entra na base. Consideram

um ponto $y = (y_1, y_2)$ tal que $y_1 = \min_{x^k}\{f_1(x^k) + \hat{c}^1_{ij}\}$. Se existirem vários pontos nestas condições consideram o que tiver menor valor para o segundo objectivo. Se este ponto não for dominado por outro anteriormente calculado e for possível enviar uma unidade de fluxo ao longo do ciclo associado, consideram este novo ponto com um conjunto de arcos, $R$, associado que tem todos os arcos do ponto que lhe deu origem menos o arco $(i, j)$. O processo é repetido até não existirem pontos com conjuntos $R$ associados não vazios.

Przybylski *et al.* (2006) mostram que o algoritmo de Sedeño Noda & González-Martín (2001) nem sempre calcula todas as soluções não-dominadas e apresentam um exemplo em que isso não acontece. Estes autores consideram um exemplo particular do problema FRIBO, em que qualquer solução admissível define um caminho de um nodo $s$ para um nodo $t$ com o fluxo em cada arco igual a 0 ou 1. Durante a execução do algoritmo, são realizadas *pivotações* que alteram o fluxo no ciclo de uma unidade. Consequentemente, cada nova solução será encontrada a partir de outra realizando uma *pivotação*. No exemplo dado, o algoritmo de Sedeño Noda & González-Martín (2001) não consegue encontrar o conjunto de todas as soluções não--dominadas, ficando por calcular uma das não-suportadas. Przybylski *et al.* (2006) concluem que não é possível determinar todas as soluções não-dominadas, para problemas de optimização combinatória bi-objectivo através de métodos baseados na *pivotação* simplex. Afirmam, ainda, que este facto já era conhecido pelo menos desde 1997.

**Figueira (2001) e Gouveia (2002)** Com base no documento Figueira (2001), Gouveia (2002) implementa um algoritmo capaz de calcular a totalidade das soluções de um problema FRIBO e identificar as eficientes, para várias variantes dos problemas de fluxo de custo mínimo: caminho mais curto, transportes, afectação, $\cdots$.

O algoritmo recorre ao algoritmo dos $k$ melhores fluxos em Hamacher (1995), para explorar de uma forma explícita a região admissível tanto no espaço dos objectivos como no espaço das variáveis de decisão, enumerando todos os fluxos admissíveis um por um e depois usando um procedimento de filtragem para extrair apenas as soluções eficientes/não-dominadas.

**Raith & Ehrgott (2009)** Raith & Ehrgott (2009) propuseram recentemente um algoritmo que determina todas as soluções não-dominadas, para um problema FRIBO. Tal como Figueira (2001) e Gouveia (2002), utilizam o algoritmo dos $k$ melhores fluxos. O algoritmo é dividido em duas fases. Na primeira fase calcula todas as soluções extremas, utilizando o algoritmo de Sedeño Noda & González-Martín (2000) modificado. Na segunda fase são determinadas as restantes soluções não-dominadas. Para isso, considera o conjunto das soluções extremas, obtidas na primeira fase, or-

denadas por ordem crescente do valor do primeiro objectivo. Cada par de soluções consecutivas, $x^i$ e $x^{i+1}$, definem a hipotenusa de um triângulo rectângulo cuja área é a região de pesquisa de possíveis soluções não-dominadas. As soluções nesta área são calculadas utilizando o algoritmo dos $k$ melhores fluxos (Hamacher, 1995) no problema definido com um único objectivo que é a função linear $\lambda_1 f_1(x) + \lambda_2 f_2(x)$, onde $\lambda_1 = f_1(x^{i+1}) - f_1(x^i)$ e $\lambda_2 = f_2(x^{i+1}) - f_2(x^i)$, que representa, por isso, a família de rectas com o mesmo declive da recta que passa por esses pontos. Com o conhecimento de mais soluções não-dominadas a área de procura de soluções não--dominadas vai sendo reduzida. Trata-se portanto de um melhoramento do algoritmo Figueira-Gouveia.

## 3.3 Algoritmos aproximados

Apesar do contributo teórico que pode trazer para o avanço da investigação nesta área, na prática a determinação de todas as soluções eficientes/não-dominadas não é adequado, não só porque o tempo para o cálculo destas soluções, assim como os recursos computacionais, são limitados, mas também porque a sua enorme quantidade tornaria quase impossível a sua interpretação, como pode ser visto em Ruhe (1988a). Atendendo a todas estas razões, os algoritmos aproximados podem ser uma ferramenta fundamental na resolução de problemas práticos.

Os estudos sobre algoritmos aproximados estão divididos em dois grupos. No primeiro grupo são considerados os algoritmos que aproximam a fronteira do conjunto das soluções não-dominadas por uma função ou conjunto de funções. Pertencem ao segundo grupo os algoritmos que calculam um número limitado de soluções, com vista a encontrarem a solução final preferida num tempo de cálculo computacional razoável. Designamos por caso contínuo o primeiro grupo e, por caso discreto, o segundo.

### 3.3.1 Caso contínuo

Todos os artigos encontrados neste grupo são para o problema com apenas dois objectivos e utilizam como base o algoritmo da sanduíche proposto por Burkard *et al.* (1991). O algoritmo da sanduíche considera o intervalo $[a, b]$ e, nas iterações seguintes, vai considerando partições, deste intervalo, cada vez com menor amplitude por forma a reduzir o erro da aproximação, terminando quando o erro for inferior ou igual a um determinado valor $\epsilon$ fixo inicialmente.

**Burkard *et al.* (1987)** Burkard *et al.* (1987) propuseram o algoritmo da sanduíche para funções convexas, $h$, definidas num intervalo limitado $[a, b] \subset \mathbb{R}$, da forma $h : [a, b] \to \mathbb{R}$. Assume-se que a função $h$ é contínua nos extremos do intervalo e que as derivadas à esquerda e à direita existem e podem ser calculados os seus valores, para qualquer $t \in ]a, b[$, e ainda que os valores das derivadas à direita em $a$ e à esquerda em $b$ são finitos. No algoritmo da sanduíche pretende-se determinar, de forma eficiente, duas funções $l(t)$ e $u(t)$ que aproximem $h(t)$ à esquerda e à direita, respectivamente, sem se afastarem muito, isto é, tal que:

$$l(t) \leqslant h(t) \leqslant u(t) \text{ e } u(t) - l(t) \leqslant \epsilon, \forall t \in [a, b],$$

para um valor positivo suficientemente pequeno de $\epsilon$. A construção de $l(t)$ e $u(t)$ é feita considerando uma divisão finita do intervalo $[a, b]$ em subintervalos $[t_i, t_{i+1}], i = 0, 1, \cdots, r-1$, onde os $r$ pontos da divisão verificam $a = t_0 < t_1 < t_2 < \cdots < t_r = b$. Para cada ponto $t_i, i = 0, 1, \cdots, r-1$ designa-se por $h_i^+$ e $h_i^-$ as derivadas de $h$ em $t_i$ à direita e esquerda, respectivamente. Então as duas funções $u(t)$ e $l(t)$ são definidas da seguinte forma:

$$u(t) = h(t_i) + \frac{h(t_{i+1}) - h(t_i)}{t_{i+1} - t_i}(t - t_i)$$

e

$$l(t) = \max\{h(t_i) + h_i^+(t - t_i), h(t_{i+1}) + h_{i+1}^-(t - t_{i+1})\},$$

para $t_i \leqslant t \leqslant t_i + 1$, $i = 0, 1, \cdots, r-1$.

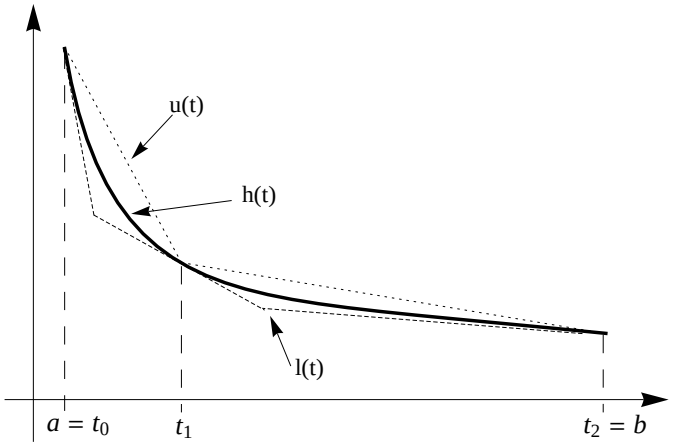



**Figura 3.1:** *Esquema ilustrativo da ideia da sanduíche.*

No início temos apenas uma divisão do intervalo $[a, b]$ ou nenhuma, se a regra do limite para o erro for verificada. Em cada iteração, um subintervalo $[t_i, t_{i+1}]$

é escolhido e um novo ponto $t^* \in ]t_i, t_{i+1}[$ é calculado, reconstruindo as funções $l(t)$ e $u(t)$. Cada novo subintervalo $[t_i, t^*]$ e $[t^*, t_{i+1}]$ que não verifique o limite do erro é adicionado a uma lista para futura divisão. Se $t^*$ é escolhido como sendo o ponto médio do intervalo $[t_i, t_{i+1}]$ dizemos que utilizamos a regra da bissecção. O erro da aproximação actual é medido em Burkard *et al.* (1991) como o valor $\max\{u(t) - l(t),\, t \in [a, b]\}$. Burkard *et al.* (1991) propõem a regra da bissecção do intervalo onde o erro é máximo.

No problema FRBO o conjunto de todas as soluções não-dominadas pode ser considerado como uma curva convexa com uma das funções, $f_1(x)$ ou $f_2(x)$, utilizada como variável independente, $t$.

Outros artigos sobre este tipo de aproximação foram publicados. As diferenças entre eles residem na forma de calcular o erro ou de calcular o novo ponto de divisão.

**Ruhe (1988b)** Ruhe (1988b) considera como medida do erro uma medida alternativa justificando a sua necessidade pelo facto da medida considerada por Burkard *et al.* (1987) variar com a rotação das funções. Ruhe considera a distância projectiva, para medir o erro, definida como

$$pdist(l(t), u(t), [a, b]) = \max\{pd1, pd2\},$$

onde $pd1 = sup_{x \in R} inf_{y \in S}\, d(x, y)$ e $pd2 = sup_{y \in S} inf_{x \in R}\, d(x, y)$, $d(x, y)$ é a distância euclidiana entre os pontos $x$ e $y$, $R = \{(t, l(t)) :\ t \in [a, b]\}$ e $S = \{(t, u(t)) :\ t \in [a, b]\}$. Além disso, é proposta uma outra regra para a divisão dos intervalos onde o limite para o erro não é verificado. De acordo com esta regra a divisão é feita no ponto da função $h(t)$ onde a linha recta de suporte é paralela à linha recta que passa pelos dois pontos extremos da curva no intervalo considerado. O algoritmo proposto funciona da seguinte forma: em primeiro lugar são determinadas soluções eficientes $x^1$ e $x^2$ que são mínimos lexicográficos da primeira e segunda funções, respectivamente. A divisão do intervalo $[a, b]$ pelos pontos $a = t_1 < t_2 < \cdots < t_r = b$ corresponde a uma divisão $0 < \lambda_1 < \cdots < \lambda_r < \infty$ do intervalo do parâmetro $\lambda$ que resolve o problema

$$\begin{array}{ll} \text{minimizar} & (c^1)^T x + \lambda (c^2)^T x, \\ \text{sujeito a:} & x \in X, \end{array}$$

para $\lambda \in ]0, \infty[$. Estes valores de $\lambda$ permitem-nos obter os valores das derivadas parciais nos pontos $t_i, i = 1, 2, \cdots, r$, que serão utilizadas na determinação de novos pontos de quebra. As primeiras funções $l(t)$ e $u(t)$ no intervalo $[a, b]$ são as seguintes:

$$u^1(t) = h(a) + \frac{h(b) - h(a)}{b - a}(t - a),$$

$$l^1(t) = \max\{h(a) + h^+(a)(t-a), h(b) + h^-(b)(t-b)\}.$$

O novo ponto, $t^*$, para a divisão do intervalo $[a,b] = [c^T x^1, c^T x^2]$ é gerado aplicando a denominada *regra da corda*, considerando $t^* \in arg\ \min\{[f(a)-f(b)]t+(b-a)h(t) : t \in [a,b]\}$. Obtemos as novas aproximações melhoradas:

$$u^2(t) = \begin{cases} h(a) + \frac{h(a)-h(t^*)}{a-t^*}(t-a), \text{ para } t \in [a,t^*] \\ h(b) + \frac{h(t^*)-h(b)}{t^*-b}(t-b), \text{ para } t \in [t^*,b] \end{cases},$$

e

$$l^2(t) = \begin{cases} h(a) + h^+(a)(t-a) & , \text{para } t \in [a,a'] \\ h(a') + \frac{h(a)-h(b)}{a-b}(t-a) & , \text{para } t \in [a',b'] \\ h(b) + h^-(b)(t-b) & , \text{para } t \in [b',b] \end{cases},$$

onde $a'$ e $b'$ são determinados pelos pontos de intersecção de $l^1(t)$ com $h(t^*) + \frac{h(a)-h(b)}{t^*-b}(t-t^*)$.

O algoritmo continua até que o erro seja inferior a um determinado valor $\epsilon$ fixo no início.

**Fruhwirth *et al.* (1989)** Fruhwirth *et al.* (1989) introduziram duas novas regras, para gerar os pontos de divisão dos intervalos, a regra da bissecção do ângulo e a regra da bissecção do declive. Na regra da bissecção do ângulo, o novo ponto $t^*$ é obtido através do ponto de tangência à curva $h(t)$, da recta com declive igual ao da bissectriz do ângulo externo das duas rectas de tangência nos pontos extremos do intervalo considerado. Na regra da bissecção do declive, o novo ponto $t^*$ é obtido através do ponto de tangência à curva $h(t)$ da recta com declive igual à média dos declives das mesmas rectas de tangência anteriores.

De estudos comparativos feitos por estes autores, chegou-se à conclusão que uma implementação especial da regra da bissecção do ângulo era a regra que se comportava melhor, quando comparada com as regras da corda e da bissecção do declive, em termos do menor consumo de tempo e memória computacionais.

**Yang & Goh (1997)** Yang & Goh (1997) propuseram uma alteração no método da bissecção modificando a aproximação inferior $l(t)$ através de uma função por troços paralelos ao da aproximação superior $u(t)$. Os novos pontos de quebra são calculados pela regra da corda. O algoritmo foi aplicado a problemas de fluxos em redes bi--objectivo quadráticos.

**Ruhe & Fruhwirth (1990)** Ruhe & Fruhwirth (1990) propõem o cálculo de uma aproximação óptima-$\epsilon$, para o problema FRBO, utilizando o algoritmo da sanduíche com a

regra de cálculo dos pontos de divisão do intervalo modificada, resolvendo, em cada iteração, dois problemas de fluxos em redes em vez de apenas um.

### 3.3.2 Caso inteiro

Sabe-se que o número de soluções eficientes/não-dominadas de um problema FRIMO é muito elevado. Como dissemos no início da Secção 3.3, o número de pontos não--dominados extremos (no espaço dos objectivos) cresce exponencialmente com o número de nodos da rede. Daí que muitos investigadores tenham optado por criar algoritmos que determinem apenas uma parte do conjunto de todas as soluções eficientes/não-dominadas ou um conjunto de pontos que possam ser uma aproximação para esse conjunto.

Do conjunto de artigos a que tivemos acesso podemos distinguir dois grupos de propostas de conjuntos aproximativos do conjunto total de soluções eficientes/não--dominadas: a) cálculo de todas as soluções eficientes/não-dominadas suportadas; b) determinação de um conjunto limitado de soluções de compromisso. No primeiro grupo estão os trabalhos de Lee & Pulat (1991), já analisado, e Nikolova (1998).

**Nikolova (1998)** Nikolova (1998) propôs um algoritmo para determinar o conjunto de todas as soluções suportadas de um problema FRIBO, com um único nodo origem, $s$, e um único nodo destino, $t$, quando uma quantidade de fluxo, $v$, é enviada de $s$ para $t$. Nikolova mostra que cada solução suportada pode ser obtida como a soma de uma solução básica admissível com fluxo de valor $v$ e um fluxo de circulação eficiente de valor zero no grafo incremental $\mathcal{G}(x)$.

**Mustafa & Goh (1998)** Mustafa & Goh (1998) propõem a utilização das soluções obtidas pelo software DINAS (Ogryczak *et al.*, 1992) para obter soluções suportadas para o problema FRIBO e para o problema correspondente com três objectivos (FRITO). O software DINAS foi concebido para a determinação de soluções não inteiras, de compromisso, para problemas de transbordo. Mustafa & Goh (1998) propõem a obtenção de soluções inteiras a partir destas. No problema FRIBO partem do pressuposto que qualquer solução não-inteira pode ser obtida através de duas soluções eficientes extremas adjacentes enviando uma quantidade de fluxo não-inteiro através do ciclo que permite passar de uma para a outra. Descobrindo qual é o ciclo, em vez de enviarmos essa quantidade não-inteira, enviamos a quantidade inteira mais próxima. O ciclo é identificado pelos arcos com fluxo não-inteiro e o sentido do ciclo é facilmente descoberto quando o desvio do fluxo no arco para o inteiro mais próximo é inferior a 1/2. Se esse desvio é igual a 1/2 é preciso traçar o ciclo para

descobrir qual o seu sentido. O resultado é depois generalizado para o caso do problema FRITO, utilizando a combinação de dois ciclos correspondentes à passagem de uma solução eficiente extrema para as eficientes extremas adjacentes. Neste caso são apresentadas algumas propriedades da combinação dos dois ciclos que permitem compreender melhor como é feito o arredondamento para o inteiro mais próximo.

**Hamacher *et al.* (2007a)** Hamacher *et al.* (2007a) propõem dois algoritmos, designados por *algoritmos das caixas*, para o cálculo de uma representação para o conjunto das soluções não-dominadas do problema FRIBO. A sua cardinalidade $N$ e a sua exactidão $\Delta$ satisfazem a relação $\mathcal{O}(\frac{A}{\Delta})$, onde $A$ é a área da caixa inicial definida pelos pontos *ideal* e *nadir*. O algoritmo começa por calcular os mínimos lexicográficos. Estes pontos são os vértices superior esquerdo e inferior direito de um rectângulo que é a caixa inicial. Este rectângulo inicial contém o conjunto de todos os pontos não-dominados. A seguir o algoritmo corta peças da caixa de partida com base em informação adicional obtida através da resolução de um problema de restrição-$\varepsilon$,

$$\begin{array}{ll} lex \text{ mínimo} & \begin{pmatrix} f_2(x) \\ f_1(x) \end{pmatrix}, \\ \text{sujeito a:} & x \in X, \\ & f_1(x) \leqslant \varepsilon, \end{array}$$

onde o valor de $\varepsilon$ é definido de duas formas diferentes dando origem a dois algoritmos. Este procedimento dá origem a um conjunto de rectângulos ou caixas que contém o conjunto dos pontos não-dominados, em cada fase do algoritmo. Além disso, para cada caixa é conhecido um ponto não-dominado que representa o conjunto das soluções não-dominadas dentro desta caixa. O critério de paragem do algoritmo baseia-se numa determinada exactidão, que é medida pela maior área dos rectângulos restantes.

**Nikolova (2001)** Nikolova (2001) propôs um algoritmo para determinar soluções eficientes para o problema FRMO quando o decisor estabelece requisitos nos limites superiores dos valores dos objectivos. Nikolova (1998) mostra que uma solução eficiente do problema FRBO, com objectivos $\sum_{i=1}^{p-1} \lambda_i f_i(x)$ e $f_p(x)$ tais que $\lambda_i \geqslant 0, i = 1, 2, \cdots, p-1$ e $\sum_{i=1}^{p-1} \lambda_i = 1$, é também eficiente para um problema FRIMO com $p$ objectivos $f_i(x), i = 1, 2, \cdots, p$. Com base neste resultado, em cada iteração o algoritmo proposto resolve um problema FRBO, cujos objectivos são a soma ponderada dos objectivos já considerados com um dos objectivos para o qual a restrição do decisor não foi preenchida. Quando um problema é não admissível é sugerida ao decisor a alteração dos limites impostos para os objectivos.

**Nikolova (2003)** Nikolova (2003) propõe outro algoritmo para determinar soluções eficientes para o problema FRMO. O algoritmo aplica regras simplificadas de classificação dos objectivos que serão utilizadas na definição de um problema cuja resolução origina uma solução eficiente para o problema FRMO inicial. O algoritmo começa por determinar uma solução eficiente utilizando o problema paramétrico, com função objectivo $\sum_{k=1}^{p} \lambda_k (\sum_{(i,j)\in\mathcal{A}} c_{ij}^k x_{ij})$, $\sum_{k=1}^{p} = 1$, utilizando pesos $\lambda_k$ iguais. Em cada iteração o utilizador deve classificar os objectivos escolhendo um que deseja melhorar, os que não se importa que sejam deteriorados, os que devem manter o seu valor e os objectivos para os quais existe indiferença. Esta classificação permite a construção de um problema bi-objectivo cuja resolução origina uma nova solução eficiente.

## 3.4 Análise e conclusões

Nesta secção fazemos a análise da literatura sobre o problema de fluxos em redes e apresentamos as nossas conclusões que justificaram o desenvolvimento deste trabalho.

O problema de fluxos em redes com vários objectivos, tanto na sua versão contínua, como na versão inteira, é um problema difícil, em primeiro lugar porque o número de soluções não-dominadas e, claro, o número de soluções eficientes pode ser um número muito elevado. Como já dissemos, Ruhe (1988a) mostrou que, para uma instância particular do problema FRBO, o número de soluções não-dominadas extremas (no espaço dos objectivos) cresce exponencialmente com o número de nodos na rede. Com esta conclusão poderíamos ser levados a esquecer qualquer trabalho na procura exacta do conjunto de todas as soluções eficientes/não-dominadas para este problema procurando apenas algoritmos aproximados. No entanto, não é possível conhecer as características destes problemas sem que se faça investigação na procura das soluções exactas e, além disso, é sabido que os algoritmos aproximados dependem, de alguma forma, da existência de algoritmos exactos, nem que seja no cálculo da estimativa do erro cometido. Assim, o trabalho no estudo de algoritmos exactos para este problema terá, certamente, uma grande importância e será de grande utilidade no futuro.

O estudo do problema de fluxos em redes com vários objectivos só muito recentemente foi iniciado. Não admira, por isso, que exista uma grande necessidade de investigação nesta área. Dos artigos revistos e salvo melhor conhecimento, apenas o artigo Ehrgott (1999) apresenta, sem sucesso, um algoritmo exacto para o problema com mais de dois objectivos, todos os restantes tratam apenas o caso com dois objectivos. No caso contínuo, para o problema FRBO, os algoritmos propostos por

Pulat *et al.* (1992) e por Sedeño Noda & González-Martín (2000) calculam todos os pontos não-dominados extremos, permitindo a seguir calcular as restantes soluções não-dominadas, como os pontos dos segmentos de recta que unem os pontos extremos associados com aquelas soluções.

No caso do problema FRIMO, é preciso considerar dois tipos de soluções não--dominadas (ver 2.7.2): suportadas e não-suportadas. Relativamente ao cálculo das soluções suportadas, Lee & Pulat (1991) na página 126 na Secção 5 denominada *Extensions* propõem uma forma de cálculo do conjunto das soluções não-dominadas suportadas para o problema FRIBO, que é a metodologia adoptada no artigo Lee & Pulat (1993) dedicado a este tipo de problemas. Este método acabou por ser utilizado como correcto em vários artigos que foram publicados a seguir. Por exemplo, Ehrgott (1999) considera que está correcto e faz a sua generalização para mais de dois objectivos. Outros artigos que fazem uso deste método são Sedeño Noda & González-Martín (2001) e Mustafa & Goh (1998). Eusébio & Figueira (2009b) apresentam um exemplo que mostra que nem sempre este método consegue calcular todas as soluções não-dominadas suportadas. No caso das soluções não-suportadas, nem todos os algoritmos propostos conseguem determinar o conjunto de todas as soluções eficientes/não-dominadas não-suportadas. Dos artigos revistos Figueira (2000), Figueira (2001), Gouveia (2002) e mais recentemente Raith & Ehrgott (2009) são os únicos capazes de determinar todo o conjunto das soluções não-dominadas/eficientes.

Dadas as críticas apontadas na revisão dos algoritmos, esta área carece de forte investigação e, por isso, pensámos em explorá-la de forma a colmatar certas falhas.

# Parte II

# Problemas lineares de fluxos em redes multi-objectivo

Na segunda parte desta Tese descrevemos o trabalho realizado sobre o problema de fluxos em redes com variáveis contínuas. Dois algoritmos primal-dual são apresentados. No Capítulo 4 descrevemos um algoritmo primal-dual para o problema FCM. O algoritmo divide o conjunto de arcos da rede em dois subconjuntos, recorrendo à informação obtida a partir do teorema dos desvios complementares, quando uma solução do problema dual é dada. Um dos subconjuntos é formado por arcos com valor do fluxo fixo igual ao limite inferior ou superior. O segundo subconjunto de arcos juntamente com um conjunto de arcos auxiliares dão origem a um problema de fluxos em redes, de que se conhece uma solução admissível inicial. A solução óptima do problema auxiliar com as variáveis associadas aos arcos no primeiro subconjunto formam uma solução que será óptima para o problema inicial, quando o valor óptimo do problema auxiliar for zero. Caso o valor da solução óptima deste problema não seja igual a zero, é gerada uma nova solução admissível para o problema dual e o procedimento é reiniciado. No Capítulo 5 descrevemos um novo algoritmo primal-dual para a determinação de todas as soluções não-dominadas extremas do problema FRBO. O algoritmo recorre a um problema paramétrico obtido a partir do problema FRBO, com o parâmetro a tomar valores no intervalo $]0,1[$. Ao longo da aplicação do algoritmo, este intervalo é dividido em subintervalos a que corresponde um problema de fluxos em redes com um único objectivo e cuja solução óptima é uma solução não-dominada extrema do problema FRBO. O algoritmo é proposto em duas versões, de acordo com a investigação realizada. Na primeira versão, a divisão do intervalo $]0,1[$ em subintervalos de menor amplitude é originada na aplicação do algoritmo primal-dual do Capítulo 4, aquando do cálculo de uma nova solução admissível para o problema dual. Verificou-se que, em geral, muitos dos subintervalos gerados iam originar a mesma solução não-dominada. Foi por esta razão que propusemos uma nova versão evitando repetições, embora continuando a utilizar a mesma estratégia de divisão do intervalo $]0,1[$, como na versão anterior. O algoritmo faz a procura de soluções não-dominadas numa árvore de divisão do intervalo $]0,1[$, explorando os subintervalos segundo um esquema de pesquisa em profundidade da esquerda para a direita. Desta forma a obtenção da primeira solução fornece conhecimento suficiente para eliminarmos os subintervalos actuais e futuros que originariam a mesma solução, reduzindo desta forma os cálculos necessários, para obtenção do conjunto pretendido. Fizemos a implementação e testamos este algoritmo. Como conclusão podemos referir que o algoritmo proposto é altamente vantajoso, em relação aos denominados paramétricos, para problemas FRBO altamente degenerados.

# Capítulo 4

# Algoritmo primal-dual para o problema de fluxo de custo mínimo

Neste capítulo apresentamos um algoritmo primal-dual para o problema FCM. Introduzimos o algoritmo e justificamos a razão de dedicarmos um capítulo a este algoritmo. Descrevemos o algoritmo nas suas diversas partes. Depois apresentamos um exemplo ilustrativo da aplicação do algoritmo. Finalmente, listamos dados e resultados da implementação e comparação deste algoritmo com o simplex para redes.

## 4.1 Introdução

Apesar de não fazer parte do plano inicial deste trabalho, este algoritmo apareceu como uma extensão natural do algoritmo Ehrgott *et al.* (2007) para problemas de programação linear bi-objectivo. Seguidamente, efectuámos estudos com vista à melhoria desse primeiro algoritmo de que resultou o algoritmo descrito no Capítulo 5. A escolha da sua descrição num capítulo isolado, em vez de uma secção do Capítulo 5, justifica-se pelo facto do trabalho aqui descrito poder ser utilizado isoladamente ou poder fazer parte de outros estudos. O problema FCM é um caso particular de um problema de programação linear e, como tal, poderia ser resolvido utilizando o algoritmo primal-dual para este tipo de problemas. No entanto, é sabido que a utilização de estruturas de representação, como é o caso da $EAG$, diferentes das matrizes podem poupar recursos computacionais como memória e tempo de cálculo (ver Bazaraa *et al.*, 2005). O algoritmo aqui proposto utiliza as condições de optimalidade dos desvios complementares (2.5.2) dividindo os arcos da rede em dois conjuntos, de acordo com o valor de uma solução dual admissível dada inicialmente ou obtida ao longo da aplicação do algoritmo. O primeiro conjunto é o conjunto de arcos da rede com fluxo, igual ao seu limite inferior ou superior, obtido através da utilização das relações dos desvios complementares e de uma solução para o problema dual. O segundo conjunto é formado pelos restantes arcos. O segundo conjunto de arcos permite construir um problema auxiliar, que é um novo problema de fluxos em redes e, como tal, resolúvel utilizando, por exemplo, o simplex para redes, ou qualquer outro algoritmo aplicado a redes.

## 4.2 Descrição do algoritmo

Consideremos o problema primal (2.1) e o seu dual (2.4). Designemos por $x$ uma solução admissível para o primal e por $(\pi, \mu)$ uma solução admissível para o dual. A última solução satisfaz as restrições do dual e, como tal, também a desigualdade $\pi_i - \pi_j - \mu_{ij} \leqslant c_{ij}$ para todo o arco $(i, j)$ na rede $\mathcal{A}$. Pelas relações dos desvios complementares (Teorema 2.5.2) se $x$ e $(\pi, \mu)$ são soluções óptimas para os respectivos problemas e $\pi_i - \pi_j \neq c_{ij}$, então ou $x_{ij} = 0$ ou $x_{ij} = u_{ij}$. Designemos por $\mathcal{A}^{=}$ o conjunto dos arcos $(i, j)$ tais que $\pi_i - \pi_j = c_{ij}$, isto é, o conjunto dos arcos com custo reduzido igual a zero. Para cada arco não pertencente a $\mathcal{A}^{=}$ façamos $x_{ij} = 0$ se $c_{ij} - \pi_i + \pi_j > 0$ e $x_{ij} = u_{ij}$ se $c_{ij} - \pi_i + \pi_j < 0$. Designemos por $L_{pd}$ o conjunto $\{(i, j) \in \mathcal{A} \setminus \mathcal{A}^{=} : x_{ij} = 0\}$ e por $U_{pd}$ o conjunto $\{(i, j) \in \mathcal{A} \setminus \mathcal{A}^{=} : x_{ij} = u_{ij}\}$. Verifica-se que $\mathcal{A} = \mathcal{A}^{=} \cup L_{pq} \cup U_{pq}$.

Consideremos o problema primal auxiliar (4.1) que resulta do primal eliminando todos os termos associados aos arcos que não pertencem a $\mathcal{A}^{=}$, adicionando $m$ va-

riáveis artificiais $y_k$, $k \in \mathcal{V}$ e substituindo a função objectivo pela soma das últimas $m-1$ variáveis artificiais.

$$
\begin{array}{ll}
\text{minimizar} & z = \sum_{k \in \mathcal{V}\setminus\{1\}} y_k, \\
\text{sujeito a:} & \sum_{(1,j)\in\mathcal{A}^=} x_{1j} - \sum_{(i,1)\in\mathcal{A}^=} x_{i1} - y_1 + \sum_{k=2}^{m} (-1)^{t_k} y_k = b'_1, \\
& \sum_{(k,j)\in\mathcal{A}^=} x_{kj} - \sum_{(i,k)\in\mathcal{A}^=} x_{ik} - (-1)^{t_k} y_k = b'_k \quad ,\forall k \in \mathcal{V}\setminus\{1\}, \\
& x_{ij} \leqslant u_{ij} \quad ,\forall (i,j) \in \mathcal{A}^=, \\
& x_{ij} \geqslant 0 \quad ,\forall (i,j) \in \mathcal{A}^=, \\
& y_k \geqslant 0 \quad ,\forall k \in \mathcal{V},
\end{array}
\tag{4.1}
$$

onde

$$
t_k = \begin{cases} 1 & \text{se } b'_k \geqslant 0 \\ 0 & \text{se } b'_k < 0 \end{cases} \text{, para } k \in \mathcal{V}.
$$

As variáveis artificiais $y_k$, $k \in \mathcal{V}$ asseguram a admissibilidade do problema (4.1). De facto, a solução com os valores $x_{ij} = 0, (i,j) \in \mathcal{A}^=$, $y_1 = 0$ e $y_k = |b'_k|,\ k \in \mathcal{V}\setminus\{1\}$ é admissível para este problema. Os valores da oferta e da procura $b'_k,\ k \in \mathcal{V}$ são obtidos de $b_k$ através da equação:

$$
b'_k = b_k + \sum_{(i,k)\in U_{pq}} x_{ik} - \sum_{(k,i)\in U_{pq}} x_{ki}.
$$

O problema primal auxiliar é um problema de fluxo de custo mínimo numa rede auxiliar. Esta rede auxiliar resulta da rede associada ao problema inicial adicionando $m-1$ arcos: $(k,1)$ (se $b'_k \geqslant 0$) ou $(1,k)$ (se $b'_k < 0$), $k \in \mathcal{V}\setminus\{1\}$. O custo associado a estes arcos é 1. A esta rede junta-se, ainda, um nodo 0 e um arco $(0,1)$ com custo nulo (ver Grigoriadis, 1986, para uma descrição mais pormenorizada). Como exemplo consideremos a rede da Figura 4.1 (a). Os arcos artificiais são os arcos mais espessos na Figura 4.1 (b). Assim, por exemplo, como $b_2 = 1$ acrescentamos o arco $(2,1)$, com custo e fluxo iguais a 1. Outro exemplo, como $b_4 = -7 < 0$, então adicionamos o arco $(1,4)$ com custo 1 e fluxo 7. Um arco $(i,j)$, inicialmente na rede, pode ou não pertencer à nova rede consoante o arco respectivo está ou não em $\mathcal{A}^=$. No caso de pertencer à nova rede o seu custo é nulo.

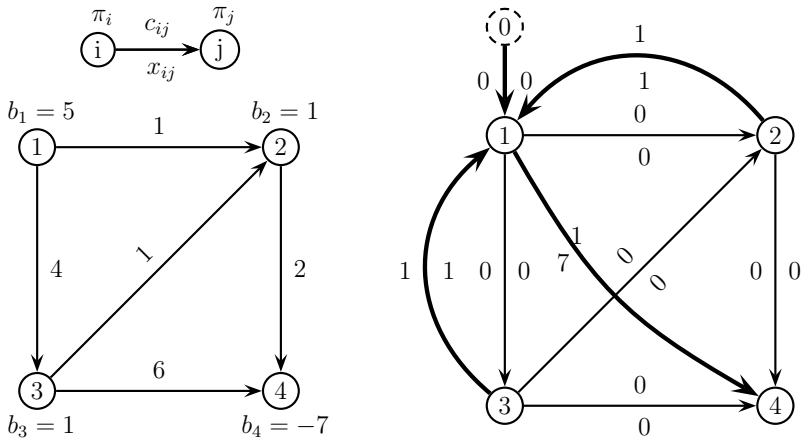



**Figura 4.1:** *(b) é uma rede auxiliar do problema de fluxos em redes (a).*

Designemos por $(\hat{x}, \hat{y})$ uma solução óptima para o problema (4.1) (pelo menos uma existe, uma vez que o problema é admissível). Se o valor óptimo $\hat{z} = \sum_{k \in \mathcal{V} \setminus \{1\}} \hat{y}_k$ é nulo, a solução $x^*$ tal que $x^*_{ij} = \hat{x}_{ij}$ para todo o arco $(i,j) \in \mathcal{A}^=$ e $x^*_{ij} = 0$ ou $x^*_{ij} = u_{ij}$ para os restantes arcos em $\mathcal{A}$ (como definido anteriormente) é uma solução admissível para o problema inicial (2.1) uma vez que todas as variáveis artificiais têm valor zero. Além disso $(\pi, \mu)$ é uma solução admissível para o dual e o par de soluções $x^*$ e $(\pi, \mu)$ verifica as relações dos desvios complementares. Sendo assim $x^*$ é uma solução óptima para (2.1).

Por outro lado, se $\hat{z} > 0$ a solução actual $\hat{x}$ não é admissível para o problema inicial (2.1). Neste caso procuramos uma nova solução para o problema dual inicial, melhor que a anterior, isto é, com um valor da função objectivo maior. Como $(\hat{x}, \hat{y})$ é uma solução óptima do problema auxiliar associado ao conjunto de arcos $\mathcal{A}^=$ devemos modificar este conjunto, adicionando um arco de forma a obter um novo problema auxiliar (4.1), com um valor objectivo inferior ou igual ao actual.

Consideremos o dual do problema (4.1)

$$\begin{array}{lrcll} \text{maximizar} & \displaystyle\sum_{k \in \mathcal{V}} b'_k \pi_k - \sum_{(i,j) \in \mathcal{A}^=} u_{ij} \mu_{ij}, & & & \\ \text{sujeito a:} & \pi_i - \pi_j - \mu_{ij} & \leqslant & 0, & \forall (i,j) \in \mathcal{A}^=, \\ & \pi_1 & \geqslant & 0, & \\ & (-1)^{t_k} \pi_1 - (-1)^{t_k} \pi_k & \leqslant & 1, & \forall k = 2, \ldots, m, \\ & \mu_{ij} & \geqslant & 0, & \forall (i,j) \in \mathcal{A}^=. \end{array} \tag{4.2}$$

Designemos por $(\hat{\pi}, \hat{\mu})$ uma solução óptima para este problema. Pelas condições

dos desvios complementares podemos escrever:

$$\begin{aligned} \hat{\pi}_i - \hat{\pi}_j - \hat{\mu}_{ij} &\leqslant 0, \\ \hat{\pi}_i - \hat{\pi}_j < 0 &\Rightarrow \hat{x}_{ij} = 0, \\ \hat{\pi}_i - \hat{\pi}_j > 0 &\Rightarrow \hat{x}_{ij} = u_{ij}, \end{aligned}$$

para todo o arco $(i,j) \in \mathcal{A}^{=}$. Consideremos agora a nova solução $(\pi', \mu')$ para o problema dual inicial (2.4) definida da seguinte forma:

$$\begin{aligned} \pi' &= \pi + \theta\hat{\pi}, \\ \mu' &= \mu + \theta\hat{\mu}, \end{aligned}$$

onde $\theta > 0$. Temos, então:

$$\begin{aligned} \pi'_i - \pi'_j - \mu'_{ij} - c_{ij} &= \pi_i + \theta\hat{\pi}_i - \pi_j - \theta\hat{\pi}_j - \mu_{ij} - \theta\hat{\mu}_{ij} - c_{ij} \\ &= \pi_i - \pi_j - \mu_{ij} - c_{ij} + \theta(\hat{\pi}_i - \hat{\pi}_j - \hat{\mu}_{ij}). \end{aligned}$$

Esta nova solução é dual admissível em (2.4) para um valor de $\theta$ suficientemente pequeno. Vejamos o que acontece com esta solução nos conjuntos $\mathcal{A}^{=}$ e no seu complementar:

1ºcaso: $(i,j) \in \mathcal{A}^{=}$

Neste caso, sabemos que $\pi_i - \pi_j - \mu_{ij} - c_{ij} \leqslant 0$ e que $\hat{\pi}_i - \hat{\pi}_j - \hat{\mu}_{ij} \leqslant 0$ então $\pi'_i - \pi'_j - \mu'_{ij} \leqslant c_{ij}$ e, portanto, a solução é dual admissível para o problema inicial.

2ºcaso: $(i,j) \notin \mathcal{A}^{=}$

Para os arcos que não fazem parte da rede auxiliar actual não conhecemos o sinal de $\hat{\pi}_i - \hat{\pi}_j - \hat{\mu}_{ij}$. Se o seu sinal for menor ou igual a zero, a nova solução é admissível qualquer que seja o valor de $\theta$. Se o seu sinal for positivo, então a admissibilidade depende do valor do $\theta$, isto é, a solução é admissível se

$$\theta \leqslant \frac{c_{ij} - \pi_i + \pi_j + \mu_{ij}}{\hat{\pi}_i - \hat{\pi}_j - \hat{\mu}_{ij}}.$$

**Observação:**

- O novo arco $(i,j)$ a ser incluído em $\mathcal{A}^{=}$ deve alterar a solução actual. Se o novo arco $(i,j)$ a juntar a $\mathcal{A}^{=}$ pertencer a $L_{pd}$ e verificar $\hat{\pi}_i - \hat{\pi}_j \leqslant 0$ ou pertencer a $U_{pd}$ e $\hat{\pi}_i - \hat{\pi}_j \geqslant 0$, a solução actual continuará óptima. Assim, consideramos os arcos em $\mathcal{A} \setminus \mathcal{A}^{=}$ que satisfazem $(i,j) \in L_{pd}$ (isto é, tais que $c_{ij} - \pi_i + \pi_j > 0$) e $\hat{\pi}_i - \hat{\pi}_j > 0$ ou $(i,j) \in U_{pd}$ (isto é, $c_{ij} - \pi_i + \pi_j < 0$) e $\hat{\pi}_i - \hat{\pi}_j < 0$.

*Algoritmo 3*: Algoritmo primal-dual.
{ *Determina um fluxo de custo mínimo.* }

---

**input:** Rede orientada $\mathcal{N} = (\mathcal{G}, c, l, u, b)$.
**output:** Solução de custo mínimo: $x$, ou resposta: problema não admissível.

(1) **begin**
(2) Determinar uma solução admissível para o dual do problema inicial: $\pi$;
(3) Construir o problema auxiliar, como em (4.1);
(4) $\theta \leftarrow 0$;
(5) **while** (o valor óptimo do problema auxiliar for positivo e existir $\theta$) **do**
(6) **begin**
(7) Determinar uma solução óptima para o dual do problema auxiliar $\hat{\pi}$;
(8) **if** $(E_{pd} \neq \emptyset)$ **then**
(9) **begin**
(10) Calcular $\theta$ como em (4.4);
(11) $\pi \leftarrow \pi + \theta\hat{\pi}$;
(12) Construir o novo problema auxiliar, como em (4.1);
(13) **end**
(14) **else** ($\theta$ não existe);
(15) **end**
(16) **if** ($\theta$ existe) **then** determinar a solução de custo mínimo $x$;
(17) **else** o problema é não admissível;
(18) **end**

---

Consideremos o seguinte conjunto:

$$E_{pd} = \{(i,j) \in \mathcal{A} : (i,j) \in L_{pd} \text{ e } \hat{\pi}_i - \hat{\pi}_j > 0 \text{ ou } (i,j) \in U_{pd} \text{ e } \hat{\pi}_i - \hat{\pi}_j < 0\} \quad (4.3)$$

Se $E_{pd} \neq \emptyset$ escolhamos $\theta > 0$ tal que:

$$\theta = \min \left\{ \frac{c_{ij} - \pi_i + \pi_j}{\hat{\pi}_i - \hat{\pi}_j} : (i,j) \in E_{pd} \right\}. \quad (4.4)$$

Deste modo, pelo menos um arco $(i,j)$, não pertencente ao conjunto actual $\mathcal{A}^=$, passará a fazer parte de $\mathcal{A}^=$ na próxima iteração. A não existência de um arco nestas condições significa que o problema dual inicial é ilimitado ou que o problema primal inicial é não admissível.

Na iteração seguinte é considerado o novo conjunto $\mathcal{A}^=$ de todos os arcos $(i,j) \in \mathcal{A}$ tais que $\pi'_i - \pi'_j = c_{ij}$ de onde resulta um novo problema auxiliar a ser resolvido.

Alguns arcos anteriormente em $\mathcal{A}^=$ podem não estar mais em $\mathcal{A}^=$ nesta nova iteração, isto é, poderão passar a fazer parte de $L_{pd}$ ou $U_{pd}$.

O processo precedente continua até $\hat{z} = 0$, onde uma solução óptima para (2.1) é obtida ou até concluirmos que o problema inicial é não admissível.

O algoritmo 3 apresenta, em resumo, os passos do algoritmo primal-dual para o problema FCM.

Na linha (2) do algoritmo, é preciso determinar uma solução inicial para o problema dual. Como usualmente, todos os custos são não-negativos, essa solução é em geral simples de obter. Por exemplo, se todos os custos forem não-negativos, $\pi = 0$ é uma solução admissível para este problema.

## 4.3 Exemplo de aplicação

Consideremos a rede ilustrada na Figura 4.2.

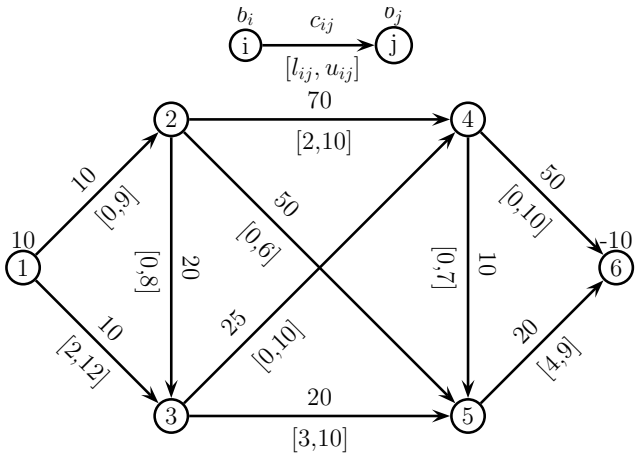



**Figura 4.2:** *Exemplo de uma rede.*

O problema de fluxo de custo mínimo associado a esta rede pode ser formulado da seguinte forma:

$$
\begin{aligned}
&\text{minimizar } 10x_{12} + 10x_{13} + 20x_{23} + 70x_{24} + 50x_{25} \\
&\qquad +25x_{34} + 20x_{35} + 10x_{45} + 50x_{46} + 20x_{56}, \\
&\text{sujeito a:} \\
&\begin{array}{rrrrrrrrrr}
x_{12} & +x_{13} & & & & & & & & = 10, \\
-x_{12} & & +x_{23} & +x_{24} & +x_{25} & & & & & = 0, \\
& -x_{13} & -x_{23} & & & +x_{34} & +x_{35} & & & = 0, \\
& & & -x_{24} & & -x_{34} & & +x_{45} & +x_{46} & = 0, \\
& & & & -x_{25} & & -x_{35} & -x_{45} & +x_{56} & = 0, \\
& & & & & & & -x_{46} & -x_{56} & = -10,
\end{array} \\
&\quad 0 \leqslant x_{12} \leqslant 9,\ 2 \leqslant x_{13} \leqslant 12,\ 0 \leqslant x_{23} \leqslant 8,\ 2 \leqslant x_{24} \leqslant 10, \\
&\quad 0 \leqslant x_{25} \leqslant 6,\ 0 \leqslant x_{34} \leqslant 10,\ 3 \leqslant x_{35} \leqslant 10,\ 0 \leqslant x_{45} \leqslant 7, \\
&\quad 0 \leqslant x_{46} \leqslant 10,\ 4 \leqslant x_{56} \leqslant 9.
\end{aligned}
$$

Este problema não está na forma geral considerada, uma vez que alguns dos limites inferiores do fluxo nos arcos são diferentes de zero e, por isso, fazemos a sua transformação no problema equivalente seguinte, onde todos os limites inferiores das variáveis são nulos, como discutido na Secção 2.2 e onde desprezamos a constante 300 da função objectivo, por não ser importante na determinação da solução. O significado de equivalência aqui é o de que ambos os problemas têm o mesmo número de soluções e as de um podem ser obtidas a partir das do outro através de relações lineares entre as suas variáveis.

$$
\begin{aligned}
&\text{minimizar } 10x_{12} + 10x_{13} + 20x_{23} + 70x_{24} + 50x_{25} \\
&\qquad +25x_{34} + 20x_{35} + 10x_{45} + 50x_{46} + 20x_{56}, \\
&\text{sujeito a:} \\
&\begin{array}{rrrrrrrrrr}
x_{12} & +x_{13} & & & & & & & & = 8, \\
-x_{12} & & +x_{23} & +x_{24} & +x_{25} & & & & & = -2, \\
& -x_{13} & -x_{23} & & & +x_{34} & +x_{35} & & & = -1, \\
& & & -x_{24} & & -x_{34} & & +x_{45} & +x_{46} & = 2, \\
& & & & -x_{25} & & -x_{35} & -x_{45} & +x_{56} & = -1, \\
& & & & & & & -x_{46} & -x_{56} & = -6,
\end{array} \\
&\quad x_{12} \leqslant 9,\ x_{13} \leqslant 10,\ x_{23} \leqslant 6,\ x_{24} \leqslant 8,\ x_{25} \leqslant 6, \\
&\quad x_{34} \leqslant 10,\ x_{35} \leqslant 7,\ x_{45} \leqslant 7,\ x_{46} \leqslant 10,\ x_{56} \leqslant 5, \\
&\quad x_{ij} \geqslant 0, \forall (i,j) \in \mathcal{A},
\end{aligned}
$$

onde $\mathcal{A} = \{(1,2),(1,3),(2,3),(2,4),(2,5),(3,4),(3,5),(4,5),(4,6),(5,6)\}$.

O dual deste problema é o seguinte:

$$\begin{aligned}
\text{maximizar } & 8\pi_1 - 2\pi_2 - \pi_3 + 2\pi_4 - \pi_5 - 6\pi_6 \\
& -9\mu_{12} - 10\mu_{13} - 6\mu_{23} - 8\mu_{24} - 6\mu_{25} \\
& -10\mu_{34} - 7\mu_{35} - 7\mu_{45} - 10\mu_{46} - 5\mu_{56}, \\
\text{sujeito a: } & \pi_1 - \pi_2 - \mu_{12} \leqslant 10, \\
& \pi_1 - \pi_3 - \mu_{13} \leqslant 10, \\
& \pi_2 - \pi_3 - \mu_{23} \leqslant 20, \\
& \pi_2 - \pi_4 - \mu_{24} \leqslant 70, \\
& \pi_2 - \pi_5 - \mu_{25} \leqslant 50, \\
& \pi_3 - \pi_4 - \mu_{34} \leqslant 25, \\
& \pi_3 - \pi_5 - \mu_{35} \leqslant 20, \\
& \pi_4 - \pi_5 - \mu_{45} \leqslant 10, \\
& \pi_4 - \pi_6 - \mu_{46} \leqslant 50, \\
& \pi_5 - \pi_6 - \mu_{56} \leqslant 20, \\
& \mu_{ij} \geqslant 0, \forall (i,j) \in \mathcal{A}.
\end{aligned}$$

Para iniciar o algoritmo precisamos de uma solução inicial para o dual. Como é fácil de ver a solução $\pi$ com todas as componentes nulas é admissível para o dual.

**It. 1:** Associados à solução dual temos os conjuntos $\mathcal{A}^{=} = \emptyset$, $L_{pd} = \mathcal{A}$ e $U_{pd} = \emptyset$. Assim, na primeira iteração todas as variáveis $x_{ij}$ são nulas e o problema auxiliar é o seguinte:

$$\begin{aligned}
\text{minimizar } & y_2 + y_3 + y_4 + y_5 + y_6, \\
\text{sujeito a: } & -y_1 + y_2 \quad +y_3 \quad -y_4 \quad +y_5 \quad +y_6 = 8, \\
& -y_2 \qquad\qquad\qquad\qquad\qquad = -2, \\
& \qquad\quad -y_3 \qquad\qquad\qquad = -1, \\
& \qquad\qquad\quad y_4 \qquad\qquad = 2, \\
& \qquad\qquad\qquad\quad -y_5 \qquad = -1, \\
& \qquad\qquad\qquad\qquad\quad -y_6 = -6, \\
& y_k \geqslant 0,\ k = 2,3,4,5,6.
\end{aligned}$$

Este problema é representado pela rede da Figura 4.3(a). O valor óptimo deste problema é diferente de zero e, por isso, a solução deste problema não é óptima para o problema inicial. O dual do problema auxiliar tem solução óptima tal que $\hat{\pi} = (0, -1, -1, 1, -1, -1)$. As diferenças $\hat{\pi}_i - \hat{\pi}_j$ para cada arco $(i,j) \in \mathcal{A}$ são apresentadas na Tabela 4.1a), assim como o valor do quociente $\frac{c_{ij} - \pi_i' + \pi_j'}{\hat{\pi}_i - \hat{\pi}_j}$. Deste modo, temos

$$\theta = \min\left\{10, 10, 5, 25\right\} = 5,$$

e a nova solução admissível para o dual inicial é

$$\pi'' = (0,0,0,0,0,0) + 5 \times (0,-1,-1,1,-1,-1) = (0,-5,-5,5,-5,-5).$$

Passamos à iteração seguinte.

**Tabela 4.1:** *Resultados obtidos nas diversas iterações.*

| arc | $c_{ij} - \pi'_i + \pi'_j$ | $\hat{\pi}_i - \hat{\pi}_j$ | $\frac{c_{ij} - \pi'_i + \pi'_j}{\hat{\pi}_i - \hat{\pi}_j}$ |
|---|---|---|---|
| $(1,2)$ | 10 | 1 | 10 |
| $(1,3)$ | 10 | 1 | 10 |
| $(2,3)$ | 20 | 0 | |
| $(2,4)$ | 70 | $-2$ | |
| $(2,5)$ | 50 | 0 | |
| $(3,4)$ | 25 | $-2$ | |
| $(3,5)$ | 20 | 0 | |
| $(4,5)$ | 10 | 2 | $\frac{10}{2} = 5$ |
| $(4,6)$ | 50 | 2 | $\frac{50}{2} = 25$ |
| $(5,6)$ | 20 | 0 | |

(a) It. 1

| arc | $c_{ij} - \pi''_i + \pi''_j$ | $\hat{\pi}_i - \hat{\pi}_j$ | $\frac{c_{ij} - \pi''_i + \pi''_j}{\hat{\pi}_i - \hat{\pi}_j}$ |
|---|---|---|---|
| $(1,2)$ | 5 | 1 | 5 |
| $(1,3)$ | 5 | 1 | 5 |
| $(2,3)$ | 20 | 0 | |
| $(2,4)$ | 80 | $-2$ | |
| $(2,5)$ | 50 | $-2$ | |
| $(3,4)$ | 35 | $-2$ | |
| $(3,5)$ | 20 | 0 | |
| $(4,5)$ | 0 | 2 | |
| $(4,6)$ | 40 | 2 | $\frac{40}{2} = 20$ |
| $(5,6)$ | 20 | 0 | $\frac{20}{2} = 10$ |

(b) It. 2

| arc | $c_{ij} - \pi'''_i + \pi'''_j$ | $\hat{\pi}_i - \hat{\pi}_j$ | $\frac{c_{ij} - \pi'''_i + \pi'''_j}{\hat{\pi}_i - \hat{\pi}_j}$ |
|---|---|---|---|
| $(1,2)$ | 0 | 0 | |
| $(1,3)$ | 0 | 0 | |
| $(2,3)$ | 20 | 0 | |
| $(2,4)$ | 90 | $-1$ | |
| $(2,5)$ | 60 | $-1$ | |
| $(3,4)$ | 45 | $-1$ | |
| $(3,5)$ | 30 | $-1$ | |
| $(4,5)$ | 0 | 0 | |
| $(4,6)$ | 30 | 2 | $\frac{30}{2} = 15$ |
| $(5,6)$ | 10 | 2 | $\frac{10}{2} = 5$ |

(c) It. 3

| arc | $c_{ij} - \pi^{iv}_i + \pi^{iv}_j$ | $\hat{\pi}_i - \hat{\pi}_j$ | $\frac{c_{ij} - \pi^{iv}_i + \pi^{iv}_j}{\hat{\pi}_i - \hat{\pi}_j}$ |
|---|---|---|---|
| $(1,2)$ | 0 | 0 | |
| $(1,3)$ | 0 | 0 | |
| $(2,3)$ | 20 | 0 | |
| $(2,4)$ | 95 | 1 | 95 |
| $(2,5)$ | 65 | 1 | 65 |
| $(3,4)$ | 50 | 1 | 50 |
| $(3,5)$ | 35 | 1 | 35 |
| $(4,5)$ | 0 | 0 | |
| $(4,6)$ | 20 | 0 | |
| $(5,6)$ | 0 | 0 | |

(d) It. 4

| arc | $c_{ij} - \pi^{v}_i + \pi^{v}_j$ | $\hat{\pi}_i - \hat{\pi}_j$ | $\frac{c_{ij} - \pi^{v}_i + \pi^{v}_j}{\hat{\pi}_i - \hat{\pi}_j}$ |
|---|---|---|---|
| $(1,2)$ | 0 | 0 | |
| $(1,3)$ | 0 | 0 | |
| $(2,3)$ | 20 | 0 | |
| $(2,4)$ | 60 | 0 | |
| $(2,5)$ | 30 | 0 | |
| $(3,4)$ | 15 | 0 | |
| $(3,5)$ | 0 | 0 | |
| $(4,5)$ | 0 | 0 | |
| $(4,6)$ | 20 | 1 | 20 |
| $(5,6)$ | 0 | $-1$ | |

(e) It. 5

| arc | $c_{ij} - \pi^{vi}_i + \pi^{vi}_j$ | $\hat{\pi}_i - \hat{\pi}_j$ | $\frac{c_{ij} - \pi^{vi}_i + \pi^{vi}_j}{\hat{\pi}_i - \hat{\pi}_j}$ |
|---|---|---|---|
| $(1,2)$ | 0 | | |
| $(1,3)$ | 0 | | |
| $(2,3)$ | 20 | | |
| $(2,4)$ | 60 | | |
| $(2,5)$ | 30 | | |
| $(3,4)$ | 15 | | |
| $(3,5)$ | 0 | | |
| $(4,5)$ | 0 | | |
| $(4,6)$ | 0 | | |
| $(5,6)$ | $-20$ | | |

(f) It. 6

**It. 2:** O arco $(4,5)$ é adicionado à rede actual uma vez que é o único arco correspondente ao valor mínimo do quociente $\frac{c_{ij}-\pi'_i+\pi'_j}{\hat{\pi}_i-\hat{\pi}_j}$.

O novo problema auxiliar está representado na Figura 4.3(b). Este problema tem solução óptima com valor da função objectivo maior que zero (ver Figura 4.3(c)). O problema dual deste tem solução óptima tal que $\hat{\pi} = (0,-1,-1,1,1,-1)$. Calculamos o valor de $\theta$, $\theta = \min\{5,5,20,10\} = 5$, que é obtido para os arcos $(1,2)$ e $(1,3)$ (ver Tabela 4.1). Assim, a nova solução dual é a seguinte:

$$\begin{aligned}\pi''' &= (0,-5,-5,5,-5,-5) + 5 \times (0,-1,-1,1,1,-1)\\ &= (0,-10,-10,10,0,-10)\end{aligned}$$

e passamos à iteração seguinte.

As iterações seguintes estão resumidas nas Tabelas 4.1 e 4.2 e nas Figuras 4.3 e 4.4. O algoritmo termina com $\pi^* = (0,-10,-10,-20,-30,-70)$, $\mathcal{A}^= = \{(1,2),(1,3),(3,5),(4,5),(4,6),(5,6)\}$, $x^* = (2,6,0,0,0,0,5,1,1,5)$ e $f(x^*) = 340$. O valor óptimo do problema inicial será, por isso, 640 com solução óptima associada $(2,8,0,2,0,0,8,1,1,9)$.

**Tabela 4.2:** *Resultados do primal-dual.*

| It. | Solução dual | $\mathcal{A}^=$ | Solução do dual auxiliar | $\theta$ |
|---|---|---|---|---|
| 1 | $\pi$=(0,0,0,0,0,0) | {} | $\hat{\pi}$=(0,-1,-1,1,-1,-1) | 5 |
| 2 | $\pi$=(0,-5,-5,5,-5,-5) | {(4,5)} | $\hat{\pi}$=(0,-1,-1,1,1,-1) | 5 |
| 3 | $\pi$=(0,-10,-10,10,0,-10) | {(1,2), (1,3), (4,5)} | $\hat{\pi}$=(0,0,0,1,1,-1) | 5 |
| 4 | $\pi$=(0,-10,-10,15,5,-15) | {(1,2), (1,3), (4,5), (5,6)} | $\hat{\pi}$=(0,0,0, -1,-1,-1) | 35 |
| 5 | $\pi$=(0,-10,-10,-20,-30,-50) | {(1,2), (1,3), (3,5), (4,5), (5,6)} | $\hat{\pi}$=(0,0,0,0,0,-1) | 20 |
| 6 | $\pi$=(0,-10,-10,-20,-30,-70) | {(1,2), (1,3), (3,5), (4,5), (4,6), (5,6)} | | |

O problema auxiliar pode ser resolvido com o algoritmo simplex para redes e é importante salientar que, em cada iteração, a resolução do problema auxiliar é feita pegando na rede óptima do problema auxiliar da iteração anterior. Desta forma, o número de iterações para a resolução do problema auxiliar é, em geral, muito reduzido (ver Figuras 4.3 e 4.4).

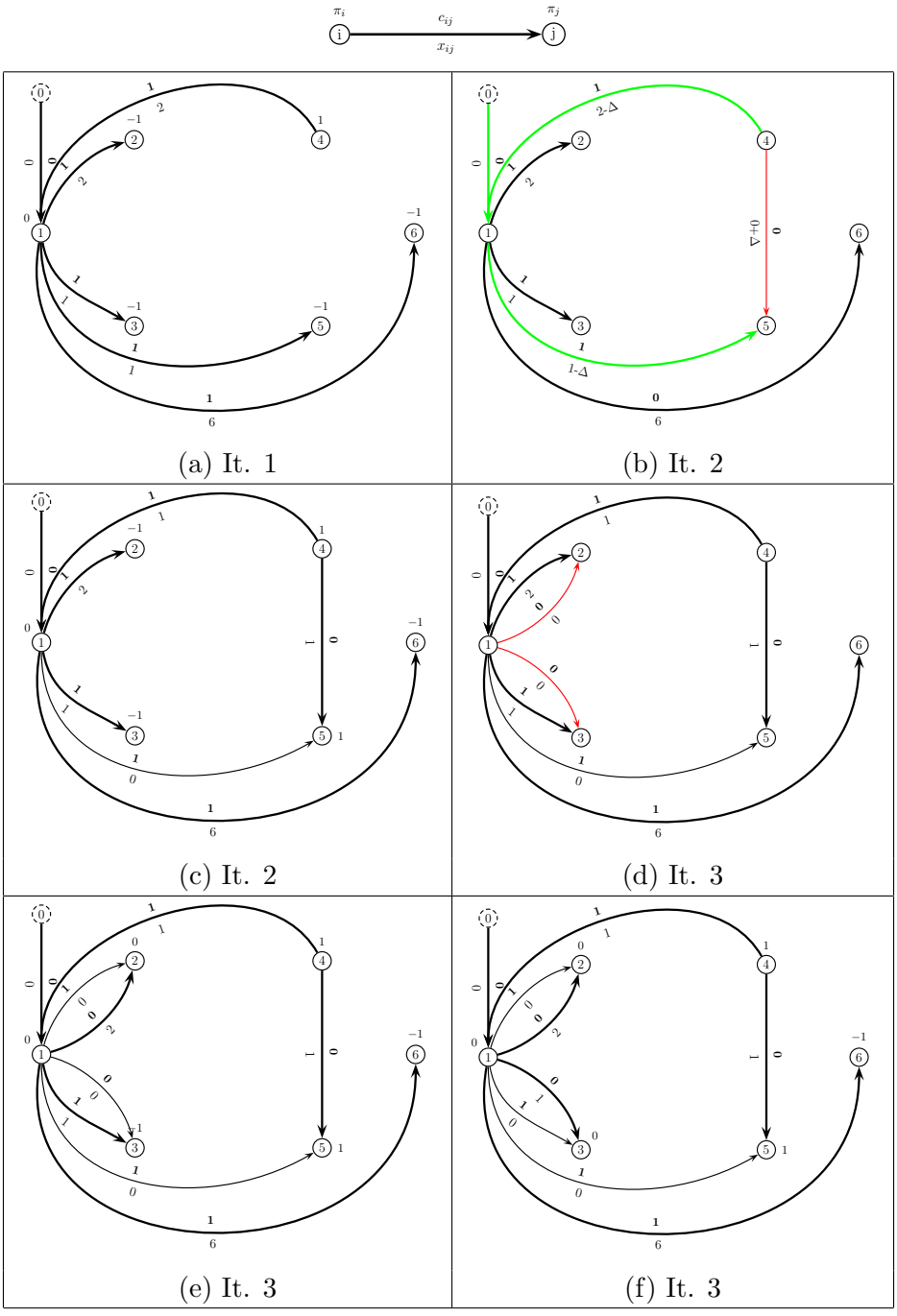
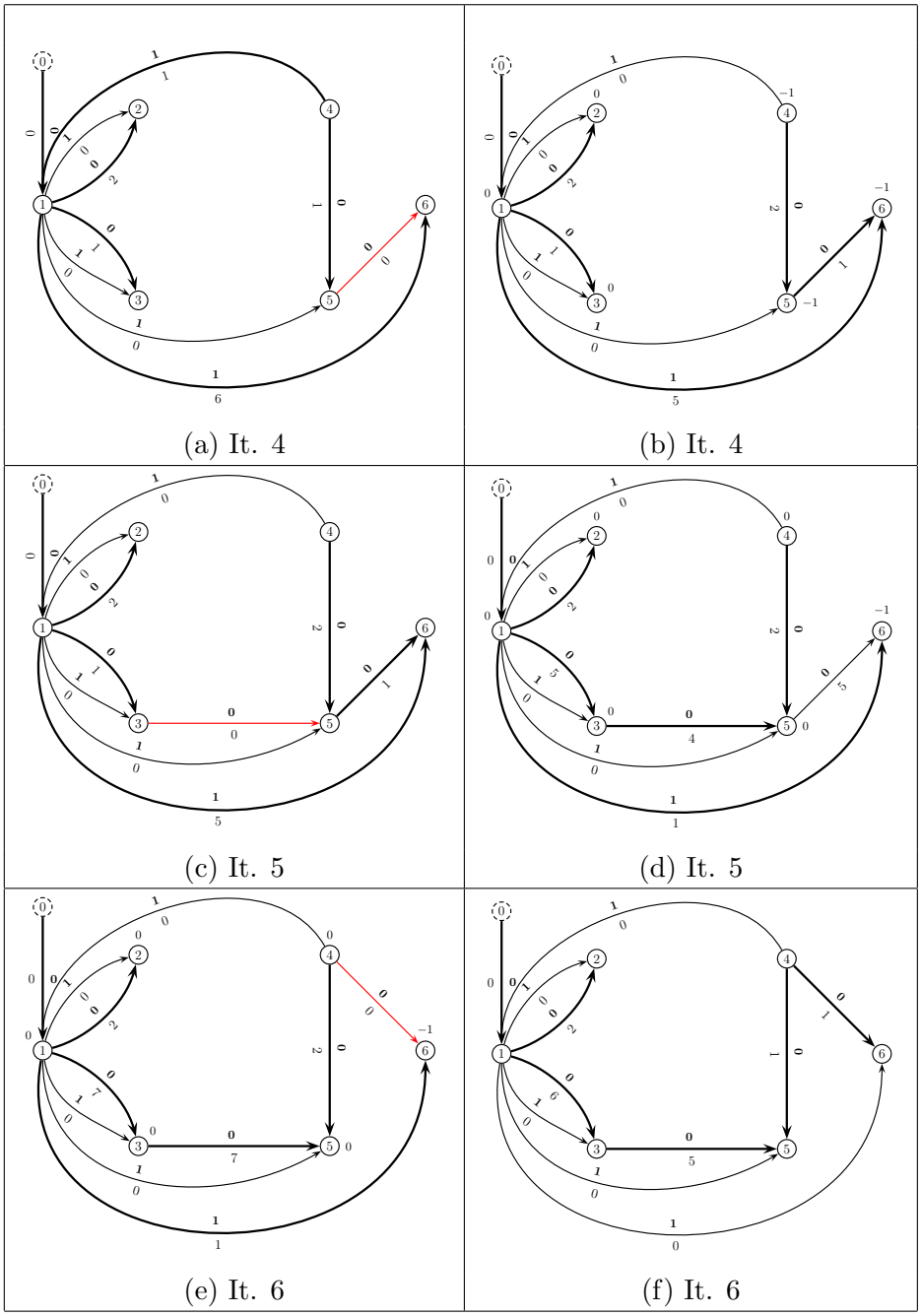



**Figura 4.3:** *Redes associadas à resolução do problema da Figura 4.2 (continua).*



(a) It. 4 (b) It. 4

(c) It. 5 (d) It. 5

(e) It. 6 (f) It. 6

**Figura 4.4:** *Redes associadas à resolução do problema da Figura 4.2 (continuação).*

## 4.4 Experiências computacionais, resultados e comentários

Implementámos ambos os algoritmos, o simplex para redes e o primal-dual descrito neste capítulo, utilizando linguagem de programação C. Testámos um conjunto de 7 problemas cada um com 30 instâncias, gerados utilizando a ferramenta *NET-GEN* (ver Klingman *et al.*, 1974). Como condição particular admitimos para todas as instâncias que os custos associados aos arcos eram não-negativos. Esta hipótese permite-nos obter uma solução admissível inicial para utilização do algoritmo primal-dual. Nestas condições, embora a diferença não fosse muito grande, o algoritmo primal-dual resolveu as instâncias geradas num tempo médio inferior ao tempo médio gasto pelo algoritmo simplex para redes, para resolver o mesmo conjunto de instâncias. Parece-nos, por isso, que podemos afirmar que este algoritmo não é pior que o simplex para redes em termos de tempo gasto, na resolução deste tipo de problemas.

**Tabela 4.3:** *Tempo CPU médio.*

| Número de nodos | número de arcos | tempo *CPU* primal-dual | tempo *CPU* simplex |
|---|---|---|---|
| 50 | 1000 | 0 | 0 |
| 60 | 1400 | 0 | 0,01 |
| 70 | 1860 | 0 | 0,01 |
| 80 | 2450 | 0,01 | 0,01 |
| 90 | 2970 | 0,01 | 0,01 |
| 100 | 4000 | 0,01 | 0,02 |
| 200 | 13090 | 0,02 | 0,04 |

# Capítulo 5

# Um algoritmo primal-dual para os problemas de fluxos em redes bi-objectivo

Neste capítulo apresentamos um algoritmo primal-dual para o problema FRBO. Fazemos uma introdução ao capítulo, descrevendo a forma como surgiu e como foi desenvolvido. Introduzimos a programação paramétrica, para problemas multi--objectivo, desenvolvendo o caso bi-objectivo, em particular, com a descrição de um algoritmo que será comparado na última secção deste capítulo com o primal-dual. Apresentamos também um exemplo de aplicação deste algoritmo. Descrevemos igualmente um novo algoritmo primal-dual para o problema FRBO. Começamos por desenvolver um algoritmo primal-dual directo e depois uma modificação deste, introduzindo melhorias na exploração do espaço de procura das soluções. Depois apresentamos um exemplo que descreve as diferenças entre as duas versões do algoritmo. Finalmente, relatamos as experiências computacionais produzidas, utilizando este algoritmo, comparamos o algoritmo paramétrico com este e apresentamos os resultados desta análise.

## 5.1 Introdução

Este capítulo apresenta um novo algoritmo primal-dual para a determinação de todas as soluções não-dominadas extremas, para o problema de fluxos em redes bi-objectivo. O algoritmo surgiu na sequência de estudos feitos para a resolução do problema FRBO utilizando o algoritmo, para problemas de programação linear multi-objectivo, apresentado em Ehrgott *et al.* (2007). Como caso particular da programação linear multi-objectivo, o problema FRBO pode ser resolvido com esse algoritmo, mas, como é sabido, a utilização de estruturas de redes em vez de matrizes melhora os recursos computacionais necessários para a resolução do problema. Por isso, numa primeira parte recriámos o algoritmo de Ehrgott *et al.* (2007), através dos conceitos da optimização em redes. O algoritmo trabalha com a análise de possíveis soluções eficientes em subintervalos do intervalo $]0, 1[$ que surgem ao longo da aplicação do algoritmo. Foi possível verificar que muitos destes subintervalos originavam a mesma solução. Surgiu, por isso, a necessidade de procurar uma alternativa que evitasse esta repetição de trabalho, isto é, a procura de soluções em subintervalos que no final originavam soluções já conhecidas. Em consequência, elaborámos uma versão modificada do algoritmo. Esta nova versão percorre o conjunto de subintervalos obtidos seguindo um esquema de pesquisa em profundidade da esquerda para a direita. Obtida a primeira solução não-dominada extrema utiliza-se o conhecimento fornecido por ela, para apagar alguns dos subintervalos já existentes e guarda-se esta informação para utilização durante a aplicação do algoritmo, levando possivelmente a que outros subintervalos sejam também apagados, com o significado que a continuação da sua análise não seria necessária pois uma solução era já conhecida. Na comparação do algoritmo proposto com um paramétrico concluímos que, em geral, o paramétrico apresenta melhores resultados computacionais, excepto para casos particulares altamente degenerados em que o algoritmo proposto é seguramente melhor.

## 5.2 Breve introdução à programação paramétrica

Uma das abordagens que aparece repetidamente na literatura, para a resolução de um problema com vários objectivos, consiste em reduzir o vector de funções objectivo a uma família de funções que resultam da soma ponderada daquelas (ver DaCunha & Polak, 1967 e Geoffrion, 1968). A abordagem funciona muito bem para o caso em que temos dois objectivos (ver Geoffrion, 1967). Na secção seguinte descrevemos como o conjunto das soluções eficientes extremas do problema FRBO pode ser obtido. O teorema seguinte apresenta uma equivalência entre o conjunto das soluções eficientes de um problema FRMO e o conjunto das soluções óptimas

de um problema paramétrico, com um único objectivo que é a soma ponderada dos objectivos do problema FRMO.

**Teorema 5.2.1 (Eficiência)** *Uma solução $x \in X$ é eficiente, para o problema FRMO, se e só se existir*

$$\lambda \in \Lambda = \Big\{ \lambda = (\lambda_1, \lambda_2, \cdots, \lambda_p) \in \mathbb{R}^p : \lambda_q > 0, q = 1, 2, \cdots, p \ e \ \sum_{q=1}^{p} \lambda_q = 1 \Big\},$$

*tal que $x$ é solução óptima do problema de programação linear*

$$\begin{array}{ll} \textit{minimizar} & \lambda^T C^T x, \\ \textit{sujeito a:} & x \in X. \end{array} \tag{5.1}$$

Este tipo de parametrização é designada por convexa. Outros tipos de parametrização podem ser encontrados na literatura (ver Steuer, 1986).

Uma *EAG* óptima para o problema (5.1) designa-se por *EAG eficiente* para o problema FRMO associado.

A seguir apresentamos uma forma de utilização do Teorema 5.2.1 para determinar todas as soluções eficientes do problema FRBO.

### 5.2.1 Caso bi-objectivo

A determinação do conjunto de todas as soluções básicas eficientes ou, alternativamente, de todas as *EAG*s eficientes, para o problema FRBO é equivalente a determinar todas as soluções óptimas de uma sequência de problemas de programação linear da forma:

$$\begin{array}{ll} \text{minimizar} & \lambda (c^1)^T x + (1-\lambda)(c^2)^T x, \\ \text{sujeito a:} & x \in X, \end{array} \tag{5.2}$$

para todo o $\lambda \in ]0,1[$ (Teorema 5.2.1). Recordando, o problema FRBO pode ser formulado como se segue:

$$\begin{array}{ll} \text{“minimizar”} & \left( (c^1)^T x, (c^2)^T x \right), \\ \text{sujeito a:} & Ax = b, \\ & 0 \geqq x \geqq u. \end{array} \tag{5.3}$$

O custo reduzido associado à função objectivo paramétrica, do problema (5.2), é:

$$\bar{c}(\lambda) = \lambda \bar{c}^1 + (1-\lambda)\bar{c}^2, \tag{5.4}$$

onde $\bar{c}^1$ e $\bar{c}^2$ representam os custos reduzidos para o primeiro e segundo objectivos, respectivamente. Consideremos uma solução básica óptima, $x$, de (5.2) para um

determinado valor $\lambda$ e uma variável não-básica $x_{ij}$ desta solução. Podemos considerar dois casos, conforme o fluxo do arco $(i,j)$ está no seu limite inferior ou no seu limite superior da seguinte forma:

A) $(i,j) \in L$, isto é, $x_{ij} = 0$.

Neste caso, atendendo a que a solução $x$ é óptima, o custo reduzido do arco $(i,j)$ é não negativo, isto é, $\bar{c}_{ij}(\lambda) \geqslant 0$, pelas condições de optimalidade (Secção 2.5). Não sabemos, no entanto, qual o sinal dos custos reduzidos deste arco quando consideramos cada um dos objectivos individualmente. Consideremos os dois casos possíveis relativamente ao primeiro objectivo: $\bar{c}^1_{ij} \geqslant 0$ ou $\bar{c}^1_{ij} < 0$ e vejamos o que acontece para cada um destes casos.

(a) $\bar{c}^1_{ij} \geqslant 0$.

De acordo com o sinal do custo reduzido, para o segundo objectivo, temos:

i. $\bar{c}^2_{ij} \geqslant 0$, o que implica $\bar{c}_{ij}(\lambda) \geqslant 0$, ou seja a condição de optimalidade para este arco é sempre verificada, qualquer que seja o valor de $\lambda$ no intervalo $]0,1[$, ou

ii. $\bar{c}^2_{ij} < 0$.
Neste caso o custo reduzido do arco $(i,j)$ é não negativo quando $\lambda \geqslant \frac{-\bar{c}^2_{ij}}{\bar{c}^1_{ij} - \bar{c}^2_{ij}}$. De facto,

$$\begin{aligned} \bar{c}_{ij}(\lambda) \geqslant 0 &\Leftrightarrow \lambda \bar{c}^1_{ij} + (1-\lambda)\bar{c}^2_{ij} \geqslant 0 \\ &\Leftrightarrow \lambda(\bar{c}^1_{ij} - \bar{c}^2_{ij}) \geqslant -\bar{c}^2_{ij} \\ &\Leftrightarrow \lambda \geqslant \frac{-\bar{c}^2_{ij}}{\bar{c}^1_{ij} - \bar{c}^2_{ij}} \end{aligned}$$

(b) $\bar{c}^1_{ij} < 0$.

Neste caso $\bar{c}^2_{ij} \geqslant 0$, obrigatoriamente, pois se $\bar{c}^2_{ij} < 0$ a solução considerada não seria óptima. Para este intervalo de valores temos,

$$\begin{aligned} \bar{c}_{ij}(\lambda) \geqslant 0 &\Leftrightarrow \lambda \bar{c}^1_{ij} + (1-\lambda)\bar{c}^2_{ij} \geqslant 0 \\ &\Leftrightarrow \lambda \leqslant \frac{-\bar{c}^2_{ij}}{\bar{c}^1_{ij} - \bar{c}^2_{ij}} \end{aligned}$$

B) $(i,j) \in U$, isto é, $x_{ij} = u_{ij}$.

Neste caso, para o arco $(i,j)$ e para este valor de $\lambda$, $\bar{c}_{ij}(\lambda) \leqslant 0$. Analisemos o que acontece para os diferentes sinais dos custos reduzidos considerando os objectivos de forma isolada.

(a) Suponhamos em primeiro lugar que $\bar{c}_{ij}^1 \leqslant 0$, então se

i. $\bar{c}_{ij}^2 \leqslant 0$, temos $\bar{c}_{ij}(\lambda) \leqslant 0$, isto é, este arco não interfere com a optimalidade da solução, qualquer que seja o valor de $\lambda$ no intervalo $]0,1[$ e se

ii. $\bar{c}_{ij}^2 > 0$, o custo reduzido da função paramétrica para o arco $(i,j)$ é não positivo quando $\lambda \geqslant \frac{-\bar{c}_{ij}^2}{\bar{c}_{ij}^1 - \bar{c}_{ij}^2}$. De facto,

$$\begin{aligned} \bar{c}_{ij}(\lambda) \leqslant 0 &\Leftrightarrow \lambda \bar{c}_{ij}^1 + (1-\lambda)\bar{c}_{ij}^2 \leqslant 0 \\ &\Leftrightarrow \lambda(\bar{c}_{ij}^1 - \bar{c}_{ij}^2) \leqslant -\bar{c}_{ij}^2 \\ &\Leftrightarrow \lambda \geqslant \frac{-\bar{c}_{ij}^2}{\bar{c}_{ij}^1 - \bar{c}_{ij}^2} \ . \end{aligned}$$

(b) Se $\bar{c}_{ij}^1 > 0$, então a solução apenas poderá ser óptima se $\bar{c}_{ij}^2 \leqslant 0$. Assim temos,

$$\begin{aligned} \bar{c}_{ij}(\lambda) \leqslant 0 &\Leftrightarrow \lambda \bar{c}_{ij}^1 + (1-\lambda)\bar{c}_{ij}^2 \leqslant 0 \\ &\Leftrightarrow \lambda \leqslant \frac{-\bar{c}_{ij}^2}{\bar{c}_{ij}^1 - \bar{c}_{ij}^2} \ . \end{aligned}$$

Consideremos os conjuntos seguintes

$$\begin{aligned} J_1 &= \{(i,j) \in L : \bar{c}_{ij}^1 \geqslant 0 \text{ e } \bar{c}_{ij}^2 < 0\} \\ &\cup \{(i,j) \in U : \bar{c}_{ij}^1 \leqslant 0 \text{ e } \bar{c}_{ij}^2 > 0\} \end{aligned} \tag{5.5}$$

e

$$\begin{aligned} J_2 &= \{(i,j) \in L : \bar{c}_{ij}^1 < 0 \text{ e } \bar{c}_{ij}^2 \geqslant 0\} \\ &\cup \{(i,j) \in U : \bar{c}_{ij}^1 > 0 \text{ e } \bar{c}_{ij}^2 \leqslant 0\}. \end{aligned} \tag{5.6}$$

Designemos por $\lambda_1$ a quantidade

$$\lambda_1 = \max\left\{ \frac{-\bar{c}_{ij}^2}{\bar{c}_{ij}^1 - \bar{c}_{ij}^2} : (i,j) \in J_1 \right\} \tag{5.7}$$

e por $\lambda_2$,

$$\lambda_2 = \min\left\{ \frac{-\bar{c}_{ij}^2}{\bar{c}_{ij}^1 - \bar{c}_{ij}^2} : (i,j) \in J_2 \right\}. \tag{5.8}$$

A solução actual do problema continua como solução óptima para o problema paramétrico quando o valor $\lambda$ pertence ao intervalo $[\lambda_1, \lambda_2]$. Este facto sugere o seguinte procedimento, para a determinação do conjunto de pontos eficientes extremos, para o problema FRBO:

1. Determinar a primeira $EAG$, $(\mathcal{T}, L, U)$, eficiente resolvendo o problema (5.2) com $\lambda$ inferior a 1, mas suficientemente próximo de 1, de tal modo que a solução seja também óptima para o problema com $\lambda \in [\lambda_1, 1]$ e $\lambda_1 < 1$. Esta solução é o mínimo lexicográfico do problema (5.3) para a primeira função objectivo.

2. Determinar o conjunto de arcos candidatos a entrar em $\mathcal{T}$ e o novo valor de $\lambda$ de acordo com (5.7) e com eles determinar as $EAG$s eficientes adjacentes.

3. Repetir o passo anterior até que todos os arcos que não pertençam a $\mathcal{T}$ sejam tais que, para $(i,j) \in L, \bar{c}^2_{ij} \geqslant 0$ e para $(i,j) \in U, \bar{c}^2_{ij} \leqslant 0$, isto é, até que a solução seja óptima relativamente ao segundo objectivo.

*Algoritmo 4*: Algoritmo paramétrico para o problema FRBO.
{*Determina o conjunto das soluções eficientes extremas.* }

---

**input:** Rede $\mathcal{N} = (\mathcal{G}, c, l, u, b)$.
**output:** Conjunto de soluções eficientes extremas: $EFE(X)$.

```
(1)  begin
(2)    Determinar uma EAG admissível/Concluir que o problema é
(3)      impossível;
(4)    Determinar o conjunto, S2, de todas as EAGs eficientes para o problema
(5)      (5.3) que são óptimas para o problema (5.2), com λ = 1;
(6)    S3 ← ∅;
(7)    while (S2 ≠ ∅) do
(8)    begin
(9)      Considerar uma EAG ρi de S2;
(10)     S2 ← S2\{ρi};
(11)     S3 ← S3 ∪ {ρi};
(12)     Determinar o conjunto S1 de EAGs eficientes adjacentes de ρi;
(13)     S2 ← S2 ∪ S1\S3;
(14)   end
(15)   Determinar o conjunto EFE(X) de todas as soluções extremas
(16)     associadas com as EAGs em S3;
(17) end
```

---

O procedimento é descrito no Algoritmo 4. Note que o problema (5.2) com $\lambda = 1$ pode ter soluções óptimas alternativas e nem todas serem eficientes para o problema (5.3). Designemos por $x$ uma solução óptima de (5.2) com $\lambda = 1$ e por $y = (y_1, y_2) = (f_1(x), f_2(x))$ a imagem de $x$ no espaço dos objectivos. Então se $x$ não é uma solução eficiente isso significa que existe uma solução eficiente $x'$ tal que $y' = (y'_1, y'_2) = (f_1(x'), f_2(x'))$, $y' \leqslant y$ com $y' \neq y$. Mas como $y_1 = \min_{x \in X} f_1(x)$ então $y'_2 < y_2$. Assim, $x'$ é uma solução óptima para (5.2) com $\lambda = 1$. Neste caso $x'$ é eficiente se e só se existir um $\lambda \in ]0, 1[$ tal que $x'$ é uma solução óptima do problema (5.2). Deste modo, existe um $0 < \theta < 1$ tal que $x'$ é uma solução óptima de (5.2) com $\lambda = 1 - \theta$. O custo reduzido para (5.2) com $\lambda = 1 - \theta$ é o seguinte:

$$\bar{c}(\lambda) = \lambda \bar{c}^1 + (1 - \lambda)\bar{c}^2 = (1 - \theta)\bar{c}^1 + \theta \bar{c}^2.$$

Se a solução $x'$ for óptima para (5.2) com $\lambda = 1$ e $\lambda = 1 - \theta$ então, relativamente a $x'$, todos os arcos $(i, j)$ em $L$ com $\bar{c}^1_{ij} = 0$ têm $\bar{c}^2_{ij} \geqslant 0$ e todos os arcos $(i, j)$ em $U$ tal que $\bar{c}^1_{ij} = 0$ têm $\bar{c}^2_{ij} \leqslant 0$.

**Exemplo 5.2.1.**
Consideremos o problema FRBO associado à rede representada na Figura 5.1.

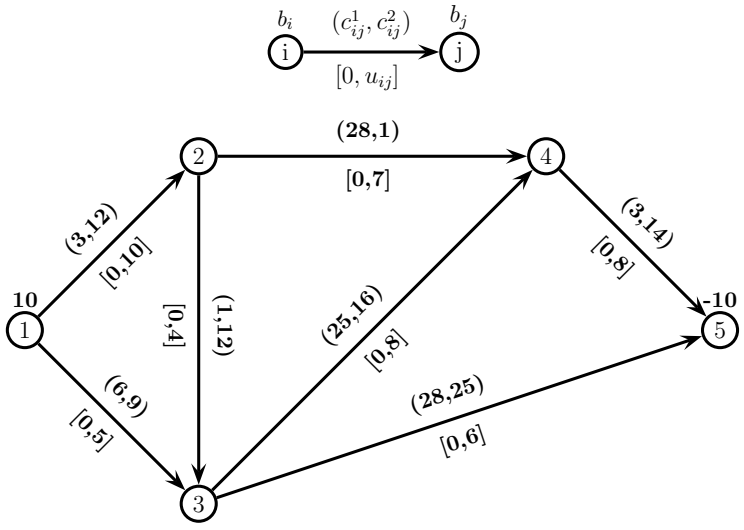



**Figura 5.1:** *Problema FRBO.*

O objectivo é encontrar todas as soluções básicas admissíveis eficientes. O primeiro passo consiste em resolver o problema FCM utilizando o primeiro objectivo.

A solução óptima é $x = (5, 5, 4, 1, 3, 6, 4)$ na Figura 5.2 ($EAG$ $\rho_{20}$), obtida utilizando o Algoritmo 1. Esta $EAG$ não é eficiente para o problema da Figura 5.1, uma vez que o fluxo da variável $x_{13}$ é igual ao limite superior do arco $(1, 3)$ e os custos reduzidos deste arco relativos ao primeiro e segundo objectivos são $\bar{c}^1_{13} = 0$ e $\bar{c}^2_{13} > 0$, respectivamente. Entre as estratégias para determinar todas as $EAG$s eficientes para o problema (5.3), que são óptimas de (5.2) com $\lambda = 1$, podem ser aplicadas as seguintes:

1. Determinar o conjunto das $EAG$s óptimas para (5.2), $S_1$, e retirar deste conjunto todas as $EAG$s não eficientes.

2. Passar da $EAG$ actual para uma adjacente até encontrar uma $EAG$ eficiente.

Consideremos a primeira estratégia. Designemos por $S_1$ o conjunto actual de $EAG$s óptimas para o problema (5.2), com $\lambda = 1$, e $S_2$ o conjunto das $EAG$s ainda não examinadas. Então $S_1 = \{\rho_{20}\}$ e $S_2 = \{\rho_{20}\}$. Façamos $S_2 = \{\}$ e determinemos as $EAG$s adjacentes de $\rho_{20}$ e óptimas para o mesmo problema. Os custos reduzidos dos arcos não-básicos na $EAG$ $\rho_{20}$ são $\bar{c}^1_{13} = 0$ e $\bar{c}^1_{35} = 0$ (ver Figura 5.2). Se o arco $(1, 3)$ ou $(3, 5)$ for o arco de entrada numa operação de *pivotação*, a solução correspondente continua óptima. Quando o arco (1,3) entra na árvore $\mathcal{T}$, $\rho_{75}$ é obtida e quando o arco $(3, 5)$ entra $\rho_{24}$ é obtida; por isso, $\rho_{75}$ e $\rho_{24}$ são $EAG$s adjacentes de $\rho_{20}$. Façamos $S_2 = \{\rho_{75}, \rho_{24}\}$. O algoritmo continua a determinar as $EAG$s adjacentes de $\rho_{75}$: $S_1 = \{\rho_{20}, \rho_{75}\}$, $S_2 = \{\rho_{24}\}$. A solução $\rho_{75}$ tem três $EAG$s adjacentes: $\rho_{20}$, $\rho^1_{91}$ e $\rho^2_{91}$. As $EAG$s $\rho^1_{91}$ e $\rho^2_{91}$ são degeneradas e representam a mesma solução. Façamos $S_2 = \{\rho^1_{91}, \rho^2_{91}\}$ e $S_1 = \{\rho_{20}, \rho_{75}, \rho_{24}\}$ e determinemos as $EAG$s adjacentes de $\rho_{24}$. Continuando desta forma até $S_2 = \{\}$ o algoritmo determina todas as $EAG$s óptimas $S_1 = \{\rho_{20}, \rho_{75}, \rho_{24}, \rho^1_{91}, \rho^2_{91}, \rho^1_{93}, \rho^2_{93}\}$ (ver Figura 5.6 e Tabela 5.1), para o problema (5.2), com $\lambda = 1$. Apenas as $EAG$s $\rho^1_{91}$ e $\rho^2_{91}$ são eficientes.

| $\rho$ | $L$ | $U$ |
|---|---|---|
| $\rho_{20}$ | {} | {(1,3),(2,3),(3,5)} |
| $\rho_{75}$ | {(3,4)} | {(2,3),(3,5)} |
| $\rho_{24}$ | {} | {(1,3),(2,3),(4,5)} |
| $\rho^1_{91}$ | {(1,3),(3,4)} | {(2,3)} |
| $\rho^2_{91}$ | {(3,4)} | {(1,2),(2,3)} |
| $\rho^1_{93}$ | {(1,3)} | {(2,3),(4,5)} |
| $\rho^2_{93}$ | {} | {(1,2),(2,3),(4,5)} |

**Tabela 5.1:** *Conjunto de soluções óptimas para o problema paramétrico com* $\lambda = 1$.

Na segunda estratégia passamos da $EAG$ óptima actual, para outra adjacente, por forma a obtermos a primeira eficiente. Não existe garantia de que este processo seja mais rápido do que o método anterior, mas, em geral este facto é verdade. A $EAG$ $\rho_{20}$ tem $\bar{c}^1_{13} = 0$ e $\bar{c}^2_{13} = 12$ e o arco $(1,3)$ pertence a $U$. Como foi visto, uma $EAG$ eficiente não pode conter um arco em $U$, com custo reduzido igual a zero para o primeiro objectivo e com custo reduzido positivo para o segundo objectivo. Então, o algoritmo determina uma $EAG$ adjacente quando o arco $(1,3)$ entra na base. A $EAG$ obtida é $\rho_{75}$ que não é eficiente, uma vez que o arco $(3,5)$ pertence a $U$ e tem custo reduzido $\bar{c}^1_{35} = 0$ e $\bar{c}^2_{35} = 7 > 0$. Inserindo o arco $(3,5)$ na árvore as $EAG$s $\rho^1_{91}$ e $\rho^2_{91}$ são obtidas através de uma operação de *pivotação*, ambas são eficientes.

Podemos agora passar ao segundo passo do algoritmo com $S_2 = \{\rho^1_{91}, \rho^2_{91}\}$. Consideremos a primeira $EAG$, $\rho^1_{91}$, representada na Figura 5.7(a). Façamos $S_2 = \{\rho^2_{91}\}$ e $S_3 = \{\rho^1_{91}\}$, e consequentemente $J_1 = \{(2,3)\}$ o que conduz a

$$\lambda_1 = \max\left\{\frac{-22}{-2-22}\right\} = \frac{11}{12}.$$

A solução actual é óptima para o problema (5.2) com $\lambda \in [\frac{11}{12}, 1]$. Quando o arco $(2,3)$ entra na árvore, é obtido o ciclo na Figura 5.7(b). Depois o arco $(2,4)$ sai da árvore e uma nova $EAG$ eficiente é obtida (ver Figura 5.7(c)), $\rho^1_{89}$. Façamos $S_2 = \{\rho^2_{91}, \rho^1_{89}\}$ e passemos à iteração seguinte.

Consideremos a $EAG$ $\rho^2_{91}$, façamos $S_2 = \{\rho^1_{89}\}$ e $S_3 = \{\rho^1_{91}, \rho^2_{91}\}$ e consequentemente $J_1 = \{(2,3)\}$ o que conduz a:

$$\lambda_1 = \max\left\{\frac{-22}{-2-22}\right\} = \frac{11}{12},$$

significando que a solução actual é óptima para o problema paramétrico com $\lambda \in [\frac{11}{12}, 1]$ e que o arco $(2,3)$ entra na árvore. O arco $(2,4)$ deixa a árvore e $\rho^2_{89}$ é obtida. Façamos $S_2 = \{\rho^1_{89}, \rho^2_{89}\}$ e passemos à iteração seguinte.

Consideremos a $EAG$ $\rho^1_{89}$, façamos $S_2 = \{\rho^2_{89}\}$ e $S_3 = \{\rho^1_{91}, \rho^2_{91}, \rho^1_{89}\}$. $J_1 = \{(1,3)\}$ e $\lambda_1 = \frac{15}{17}$. Consequentemente, o arco $(1,3)$ entra na árvore e uma nova $EAG$ é obtida, $\rho_{47}$ (ver Figura 5.7(e)). Façamos $S_2 = \{\rho^2_{89}, \rho_{47}\}$ e passemos à iteração seguinte.

Consideremos a $EAG$ $\rho^2_{89}$, façamos $S_2 = \{\rho_{47}\}$ e $S_3 = \{\rho^1_{91}, \rho^2_{91}, \rho^1_{89}, \rho^2_{89}\}$. $J_1 = \{(1,2)\}$ e $\lambda_1 = \frac{15}{17}$. A solução $\rho_{47}$ obtida já está em $S_2$. Passemos à iteração seguinte.

Consideremos a $EAG$ $\rho_{47}$, façamos $S_2 = \{\}$ e $S_3 = \{\rho^1_{91}, \rho^2_{91}, \rho^1_{89}, \rho^2_{89}, \rho_{47}\}$. $J_1 = \{\}$. Como $S_2 = \{\}$ o ciclo *while* termina.

O conjunto de soluções eficientes extremas é o seguinte:

$$\begin{aligned} EFE(X) &= \{(10,0,4,6,0,4,6), \\ &\quad (10,0,3,7,0,3,7), \\ &\quad (7,3,0,7,0,3,7)\}. \end{aligned}$$

■

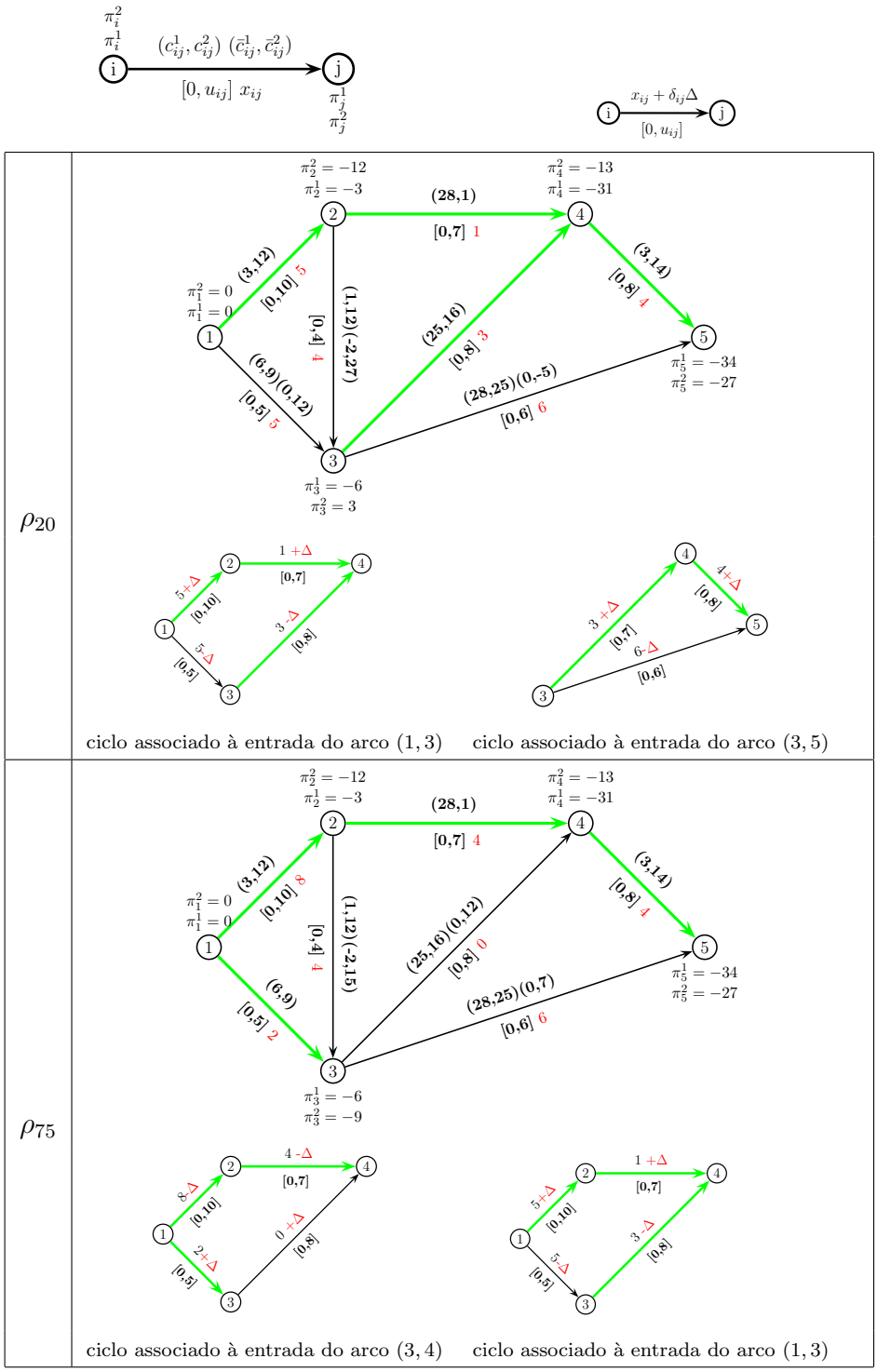



**Figura 5.2:** *Soluções óptimas do FCM do exemplo 1, considerando apenas o primeiro objectivo.*

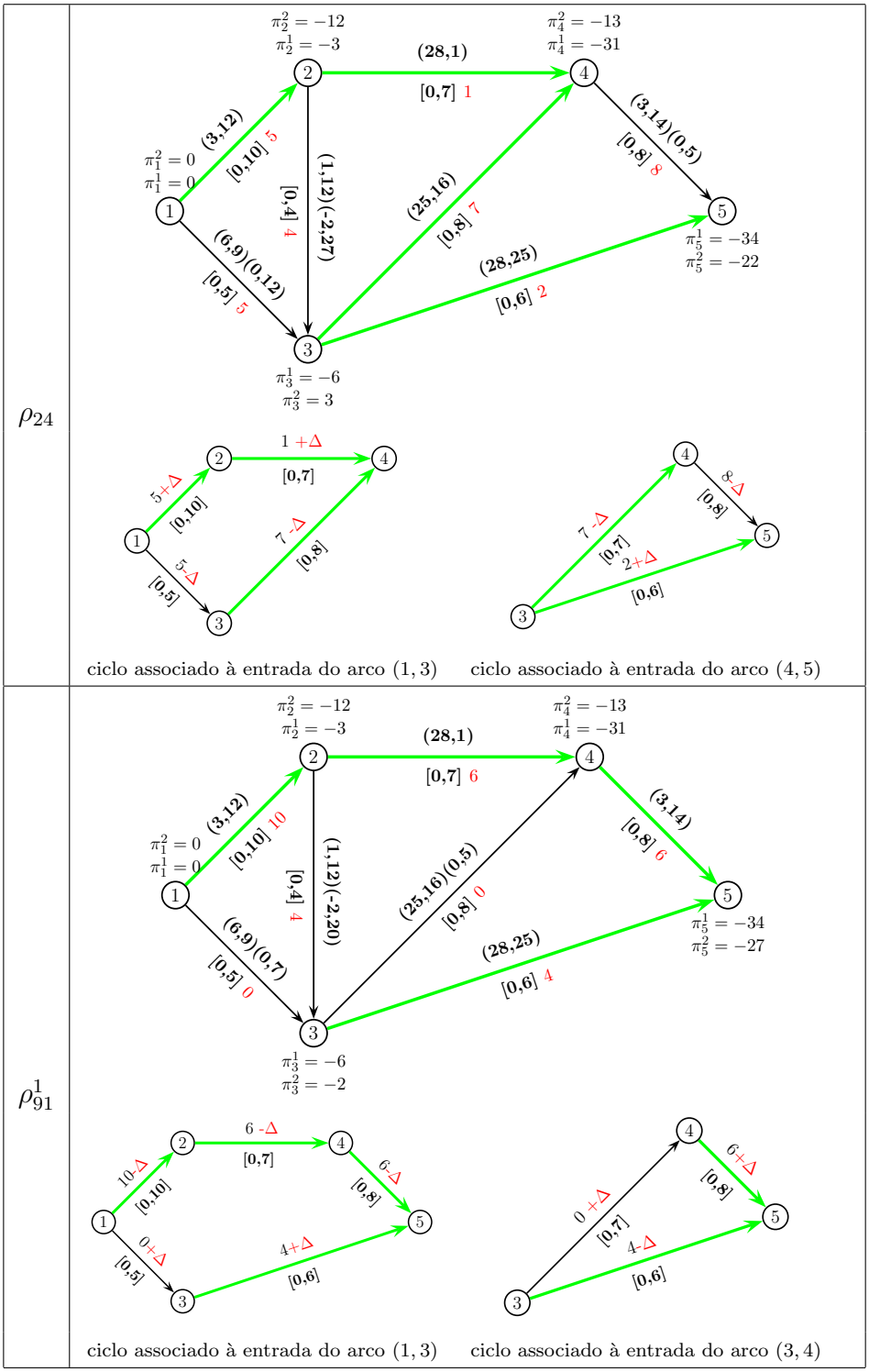



**Figura 5.3:** *Soluções óptimas do FCM do exemplo 1 (continuação).*

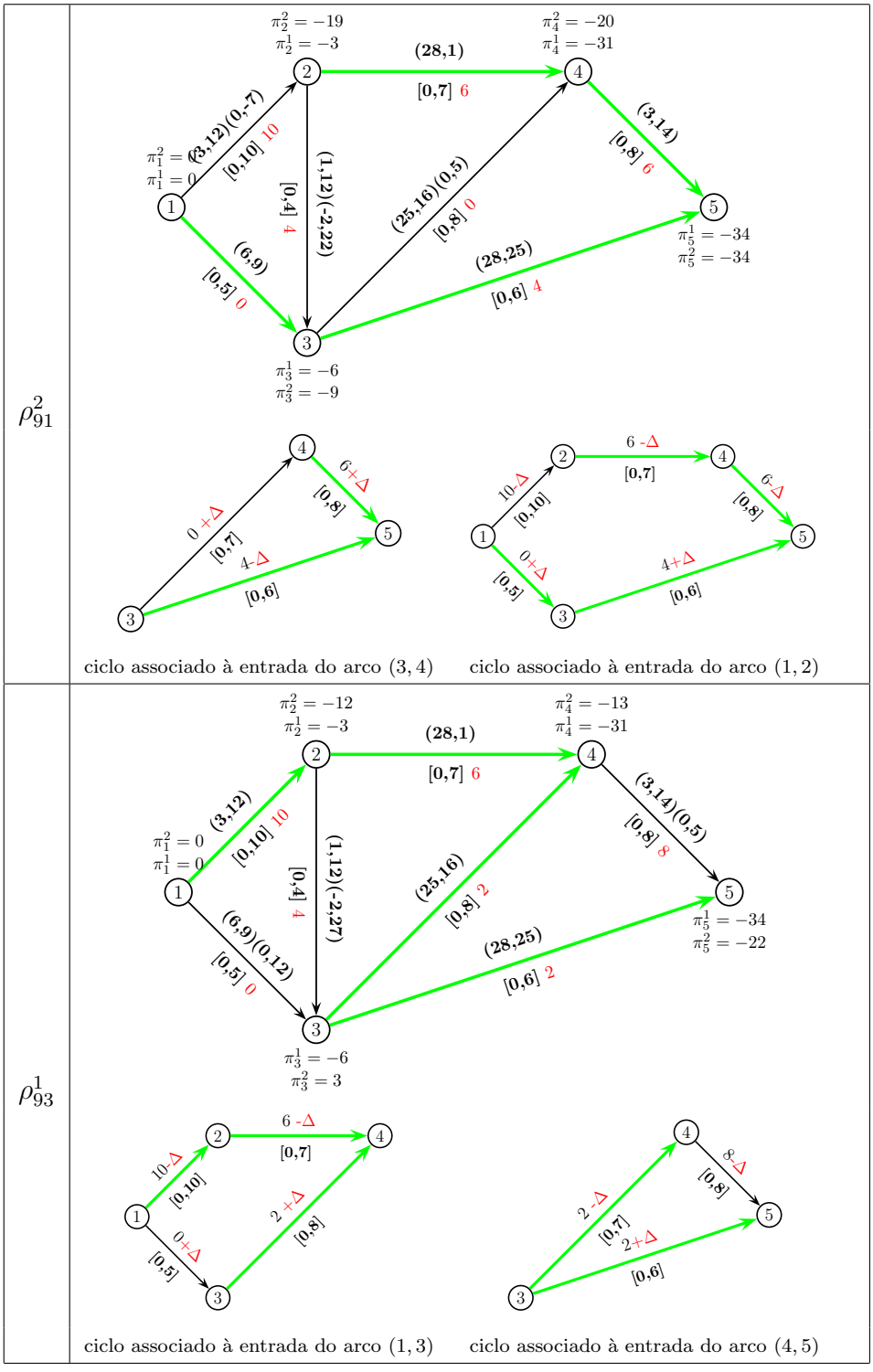



**Figura 5.4:** *Soluções óptimas do FCM do exemplo 1 (continuação).*

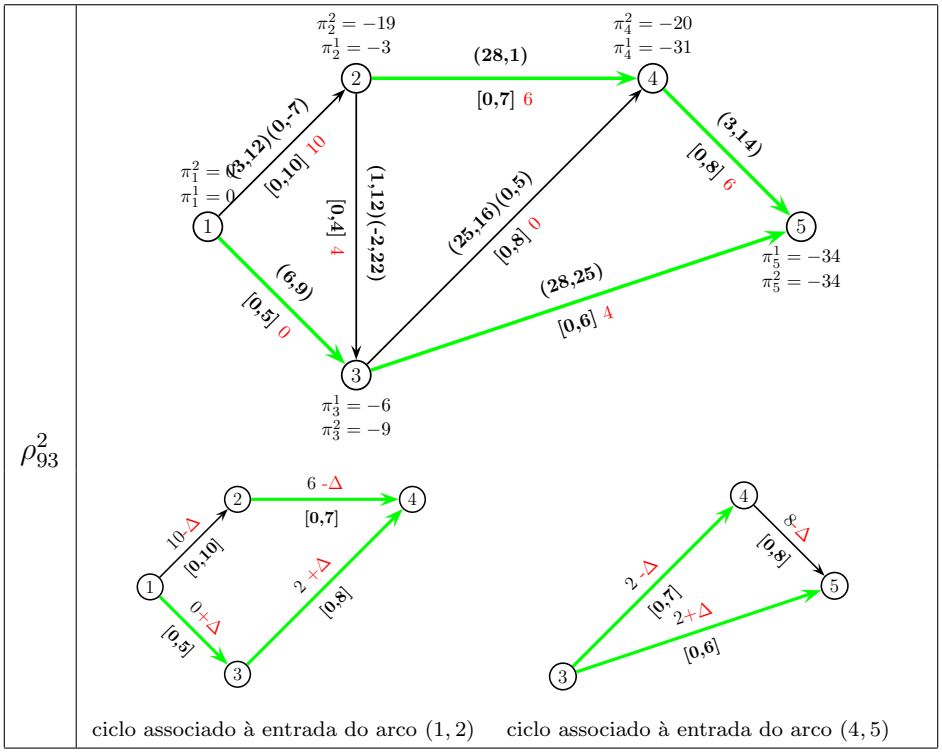



**Figura 5.5:** *Soluções óptimas do FCM do exemplo 1 (continuação).*

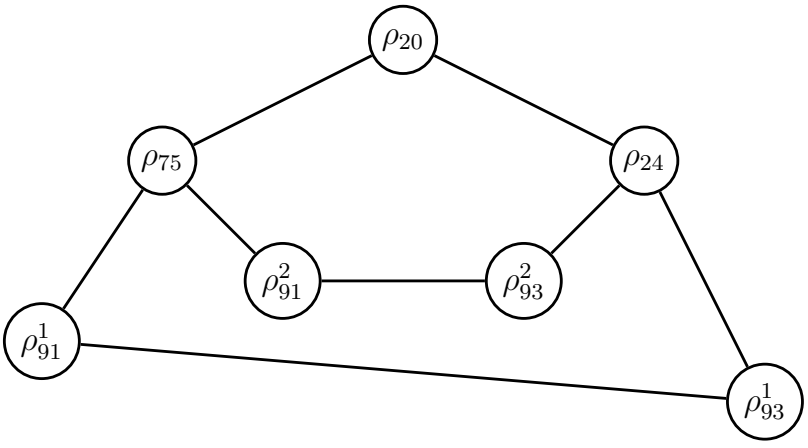



**Figura 5.6:** *Ligação entre as EAGs óptimas do exemplo 1, usando apenas o primeiro objectivo.*

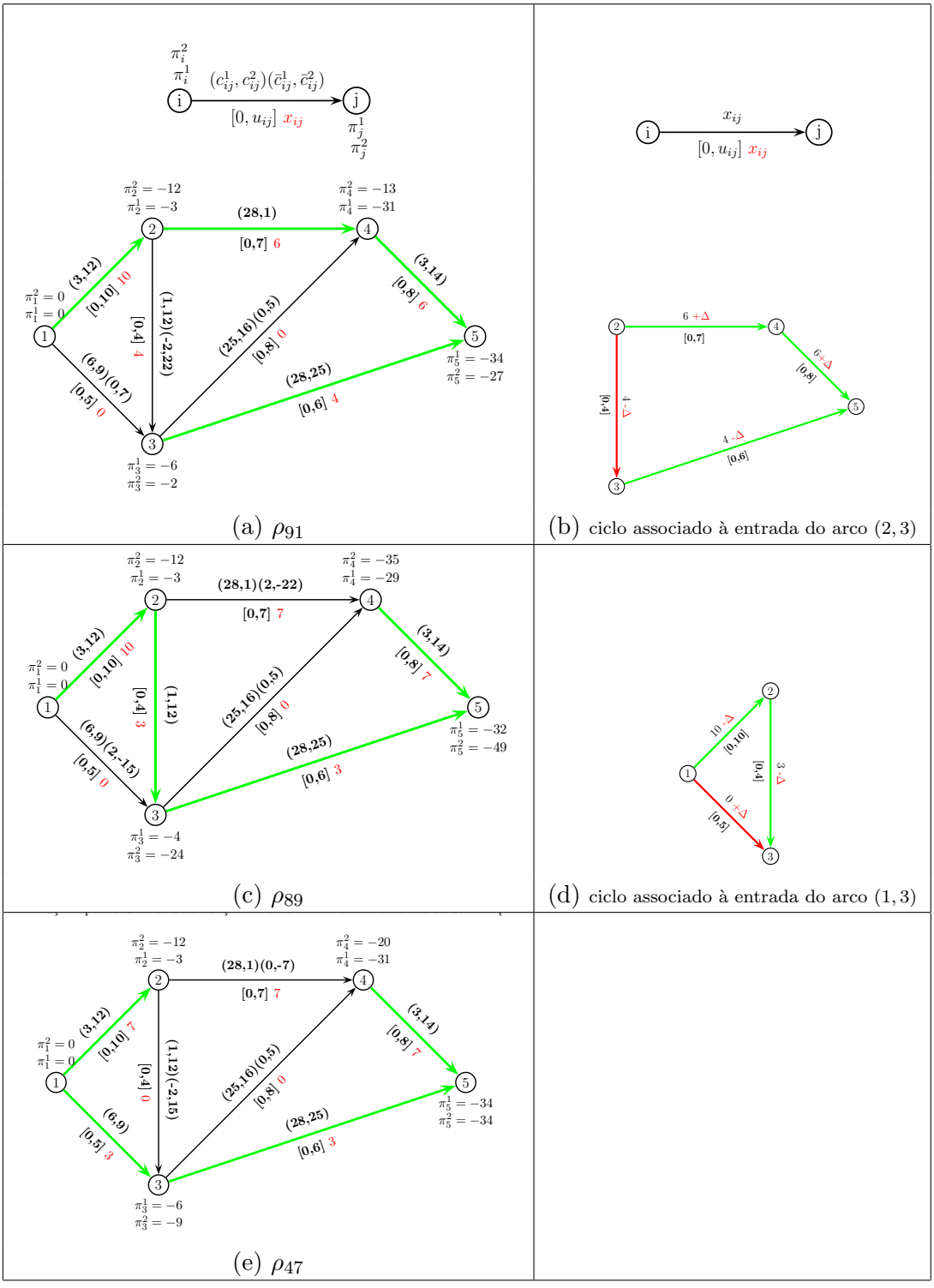



**Figura 5.7:** *EAGs eficientes.*

## 5.3 Algoritmo primal-dual

Nesta secção introduzimos um novo algoritmo primal-dual bi-objectivo, para o cálculo de todos os pontos não-dominados extremos, no espaço dos objectivos. A ideia baseia-se no trabalho de Ehrgott *et al.* (2007), onde é apresentado um algoritmo primal-dual para o problema de programação linear multi-objectivo. Construímos, em primeiro lugar, um algoritmo primal-dual para determinar todos os pontos não--dominados extremos, no espaço dos objectivos. Ao testar este algoritmo, verificámos que a mesma solução extrema poderia ser obtida em diferentes iterações do algoritmo. Evitar este tipo de repetição foi, por isso, um dos nossos objectivos. Para isso, propomos uma modificação do algoritmo primal-dual elaborado inicialmente, tirando partido da informação obtida em cada solução calculada, evitando assim a obtenção de soluções repetidas.

### 5.3.1 Descrição do algoritmo

O problema FRBO pode ser também formulado da seguinte forma:

$$\begin{array}{lll} \text{minimizar} & f_1(x) = (c^1)^T x = \sum\limits_{(i,j)\in\mathcal{A}} c^1_{ij} x_{ij}, & \\ \text{minimizar} & f_2(x) = (c^2)^T x = \sum\limits_{(i,j)\in\mathcal{A}} c^2_{ij} x_{ij}, & \\ \text{sujeito a:} & \sum\limits_{(k,j)\in\mathcal{A}} x_{kj} - \sum\limits_{(i,k)\in\mathcal{A}} x_{ik} = b_k, & \forall k \in \mathcal{V}, \\ & x_{ij} \geqslant 0, & \forall (i,j) \in \mathcal{A}, \\ & x_{ij} \leqslant u_{ij}, & \forall (i,j) \in \mathcal{A}. \end{array} \tag{5.9}$$

Designemos por $X$ o conjunto de *soluções admissíveis* para o problema (5.9) no espaço das variáveis de decisão $\mathbb{R}^n$ e por $Y = \{y = (y_1, y_2) \in \mathbb{R}^2 : y_1 = f_1(x),\ y_2 = f_2(x), x \in X\}$ o conjunto das soluções admissíveis no *espaço dos objectivos* $\mathbb{R}^2$. Como hipótese, consideramos que o problema tem pelo menos duas soluções não-dominadas diferentes, isto é, que os mínimos lexicográficos são diferentes.

De acordo com o Teorema 5.2.1, a determinação de todas as soluções eficientes de (5.9) é equivalente à determinação de todas as soluções óptimas do conjunto de problemas de fluxos em redes da forma

$$\begin{array}{lll} \text{minimizar} & \sum\limits_{(i,j)\in\mathcal{A}} \left(\lambda(c^1_{ij} - c^2_{ij}) + c^2_{ij}\right) x_{ij}, & \\ \text{sujeito a:} & \sum\limits_{(k,j)\in\mathcal{A}} x_{kj} - \sum\limits_{(i,k)\in\mathcal{A}} x_{ik} = b_k, & \forall k \in \mathcal{V}, \\ & x_{ij} \leqslant u_{ij}, & \forall (i,j) \in \mathcal{A}, \\ & x_{ij} \geqslant 0, & \forall (i,j) \in \mathcal{A}, \end{array} \tag{5.10}$$

para todos os valores de $\lambda \in ]0,1[$.

Como $X$ e $Y$ são poliedros e uma vez que o conjunto das soluções não-dominadas de $Y$ é um subconjunto da fronteira de $Y + \mathbb{R}^2_{\geqq} = \mathbb{R}^p_{\geqq}\{y + d : y \in Y, d \in \mathbb{R}^2, d \geqq 0\}$, $Y$ tem um número finito de pontos extremos $\{y^1, \ldots, y^t\}$. Assumiremos que as soluções estão indexadas de tal forma que $y^i_1 < y^{i+1}_1$ (e, deste modo $y^i_2 > y^{i+1}_2$) para $i = 1, \ldots, t-1$. Todas as outras soluções não-dominadas são combinações convexas de $y^i$ e $y^{i+1}$ para algum $i \in \{1, \ldots, t-1\}$. Para resolver o problema de fluxos em redes bi-objectivo é por isso suficiente encontrar, para cada $i = 1, \ldots, t$ uma solução admissível $x^i$ de (5.9) tal que $f_1(x^i) = y^i_1$ e $f_2(x^i) = y^i_2$.

Além disso, sabemos que existe uma divisão finita do intervalo $]0,1[$ em subintervalos tal que uma solução eficiente extrema de (5.9) está associada com um e apenas um dos subintervalos dessa divisão (ver Benson & Sun, 2000). O algoritmo de Ehrgott *et al.* (2007) resolve um programa linear multi-objectivo aplicando o algoritmo primal-dual ao problema paramétrico, como no Teorema 5.2.1, de forma a construir a divisão do intervalo $]0,1[$ e calcula uma solução eficiente para cada subintervalo desta divisão. A mesma ideia é usada aqui para encontrar todas as soluções não-dominadas extremas para o problema de fluxos em redes bi-objectivo (5.9). Consideremos o problema (5.10). O seu dual é o seguinte:

$$\begin{array}{ll} \text{maximizar} & \displaystyle\sum_{k\in\mathcal{V}} b_k\pi_k - \sum_{(i,j)\in\mathcal{A}} u_{ij}\mu_{ij}, \\ \text{sujeito a:} & \pi_i - \pi_j - \mu_{ij} \leqslant \lambda(c^1_{ij} - c^2_{ij}) + c^2_{ij},\ \forall (i,j) \in \mathcal{A}, \\ & \mu_{ij} \geqslant 0,\ \forall (i,j) \in \mathcal{A}, \\ & \pi_k \in \mathbb{R},\ \forall k \in \mathcal{V}. \end{array} \tag{5.11}$$

Como o segundo membro das restrições deste problema depende de $\lambda$, as soluções admissíveis também dependem de $\lambda$. Designemos por $(\pi(\lambda), \mu(\lambda))$ uma solução admissível de (5.11) e por $\Lambda_q, q = 1, 2, \ldots, r$ uma divisão de $]0,1[$, tal que, para cada $q$, existe $\mathcal{A}^=_q \subset \mathcal{A}$ e tal que, para todo o $\lambda \in \Lambda_q$:

1. $\pi_i(\lambda) - \pi_j(\lambda) = \lambda(c^1_{ij} - c^2_{ij}) + c^2_{ij} \quad \forall (i,j) \in \mathcal{A}^=_q$ e
2. $\pi_i(\lambda) - \pi_j(\lambda) \neq \lambda(c^1_{ij} - c^2_{ij}) + c^2_{ij} \quad \forall (i,j) \notin \mathcal{A}^=_q$,

e para todo o $\lambda \in ]0,1[\backslash\Lambda_q$ temos $\pi_i(\lambda) - \pi_j(\lambda) \neq \lambda(c^1_{ij} - c^2_{ij}) + c^2_{ij}$, para algum $(i,j) \in \mathcal{A}^=_q$.

Consequentemente, para $\lambda \in \Lambda_q$ e $(i,j) \in \mathcal{A} \setminus \mathcal{A}^=_q$ devemos ter $x_{ij} = 0$ ou $x_{ij} = u_{ij}$. Designemos por $L_q$ o conjunto de arcos $(i,j)$ tal que $x_{ij} = 0$ (assim $\pi_i(\lambda) - \pi_j(\lambda) < \lambda(c^1_{ij} - c^2_{ij}) + c^2_{ij}$ qualquer que seja $(i,j) \in L_q$) e por $U_q$ o conjunto de arcos $(i,j)$ tal que $x_{ij} = u_{ij}$ (assim $\pi_i(\lambda) - \pi_j(\lambda) > \lambda(c^1_{ij} - c^2_{ij}) + c^2_{ij}$ qualquer que seja $(i,j) \in U_q$).

Para cada intervalo $\Lambda_q$ necessitamos resolver um problema primal auxiliar, como em (4.1):

$$
\begin{array}{ll}
\text{minimizar} & z = \sum\limits_{k\in\mathcal{V}\setminus\{1\}} y_k, \\
\text{sujeito a:} & \sum\limits_{(1,j)\in\mathcal{A}_q^=} x_{1j} - \sum\limits_{(i,1)\in\mathcal{A}_q^=} x_{i1} - y_1 + \sum\limits_{k=2}^{m}(-1)^{t_k} y_k = b'_1, \\
& \sum\limits_{(k,j)\in\mathcal{A}_q^=} x_{kj} - \sum\limits_{(i,k)\in\mathcal{A}_q^=} x_{ik} - (-1)^{t_k} y_k = b'_k,\ \forall k \in \mathcal{V}\setminus\{1\}, \\
& x_{ij} \leqslant u_{ij},\ \forall (i,j) \in \mathcal{A}_q^=, \\
& x_{ij} \geqslant 0,\ \forall (i,j) \in \mathcal{A}_q^=, \\
& y_k \geqslant 0,\ \forall k \in \mathcal{V}.
\end{array}
\qquad (AP(\mathcal{A}^=))
$$

Note que $(AP(\mathcal{A}^=))$ não depende de $\lambda$. Assim, se o valor óptimo do problema $(AP(\mathcal{A}^=))$, $\hat{z}$, for zero a solução óptima $x^*$ formada pelas componentes desta solução a que juntamos as restantes componentes em $L_q$ e $U_q$ é também óptima para (5.10), para qualquer $\lambda \in \Lambda_q$, tal como acontecia no algoritmo primal-dual com um único objectivo. Se $\hat{z} > 0$ podemos formular o dual $(DAP(\mathcal{A}^=))$ da seguinte forma:

$$
\begin{array}{ll}
\text{maximizar} & \sum\limits_{k\in\mathcal{V}} b'_k \pi_k - \sum\limits_{(i,j)\in\mathcal{A}_q^=} u_{ij}\mu_{ij}, \\
\text{sujeito a:} & \pi_i - \pi_j - \mu_{ij} \leqslant 0,\ \forall (i,j) \in \mathcal{A}_q^=, \\
& \pi_1 \geqslant 0, \\
& (-1)^{t_k}\pi_1 - (-1)^{t_k}\pi_k \leqslant 1,\ \forall k = 2,\ldots,m, \\
& \mu_{ij} \geqslant 0,\ \forall (i,j) \in \mathcal{A}_q^=.
\end{array}
\qquad (DAP(\mathcal{A}^=))
$$

Designemos por $(\hat{\pi}, \hat{\mu})$ uma solução óptima do problema $(DAP(\mathcal{A}^=))$. Uma nova solução para o problema dual inicial (5.11) é então determinada da seguinte forma:

$$
\begin{aligned}
\pi'(\lambda) &= \pi(\lambda) + \theta(\lambda)\hat{\pi}, \\
\mu'(\lambda) &= \mu(\lambda) + \theta(\lambda)\hat{\mu},
\end{aligned}
$$

onde $\theta(\lambda) > 0$ é definido por,

$$
\theta(\lambda) = \min\Big\{ \tfrac{\lambda(c^1_{ij}-c^2_{ij})+c^2_{ij}-\pi_i+\pi_j}{\hat{\pi}_i-\hat{\pi}_j} :\ \begin{array}{r}((i,j)\in L_q \text{ e } \hat{\pi}_i-\hat{\pi}_j>0) \text{ ou} \\ ((i,j)\in U_q \text{ e } \hat{\pi}_i-\hat{\pi}_j<0)\Big\}.\end{array}
\qquad (5.12)
$$

Note que $\theta(\lambda)$ é uma função de $\lambda$ uma vez que o quociente $(\lambda(c^1_{ij}-c^2_{ij})+c^2_{ij}-\pi_i+\pi_j)/(\hat{\pi}_i-\hat{\pi}_j)$ depende de $\lambda$. Assim, o intervalo $\Lambda_q$ é dividido em $r' \geqslant 0$ intervalos $\Lambda_{qq'}, q' = 1,2,\ldots,r'$. Para cada um destes subintervalos, um novo conjunto de arcos $\mathcal{A}^=_{qq'}$ é definido e o procedimento é repetido. O Algoritmo 5 esquematiza as operações do método que acabámos de descrever.

*Algoritmo 5*: Algoritmo primal-dual bi-objectivo.
{*Determina o conjunto das soluções não-dominadas extremas.* }

---

**input:** Rede orientada $\mathcal{N} = (\mathcal{G}, (c^1, c^2), l, u, b)$.
**output:** Conjunto de soluções não-dominadas extremas: $NDE(Y)$.

(1) **begin**
(2) $\Lambda_0 \leftarrow ]0, 1[$, $\pi^0 \leftarrow 0$ e $NDE(Y) \leftarrow \emptyset$;
(3) $\mathcal{L} \leftarrow \{(\Lambda_0, \mathcal{A}_0^{=} = \emptyset, L_0 = \mathcal{A}, U_0 = \emptyset, \pi^0 = 0)\}$;
(4) **while** $(\mathcal{L} \neq \emptyset)$ **do**
(5) **begin**
(6) $l \leftarrow (\Lambda_q, \mathcal{A}_q^{=}, L_q, U_q, \pi^q) \in \mathcal{L}$;
(7) $\hat{y} \leftarrow$ valor óptimo de $AP(\mathcal{A}_q^{=})$;
(8) **begin**
(9) **if** $(\hat{y} = 0)$ **then**
(10) **begin**
(11) $\mathcal{L} \leftarrow \mathcal{L} \setminus \{l\}$;
(12) $y^* \leftarrow$ imagem da solução do primal associada a $\hat{y} = 0$;
(13) $NDE(Y) \leftarrow NDE(Y) \cup \{y^*\}$;
(14) **end**
(15) **else**
(16) **begin**
(17) $\hat{\pi} \leftarrow$ valor óptimo de $DAP(\mathcal{A}_q^{=})$;
(18) Determinar a divisão $\Lambda_{qq'}$, $q' = 1, \ldots, r'$ de $\Lambda_q$ de acordo com
(19) (5.12) e a nova solução dual $\pi^{qq'}(\lambda)$ qualquer que seja $q' =$
(20) $1, \ldots, r'$;
(21) $\mathcal{L} \leftarrow \mathcal{L} \setminus \{l\} \cup \{(\Lambda_{qq'}, L_{qq'}, U_{qq'}, \pi^{qq'}(\lambda), \mu^{qq'}(\lambda)) : q' = 1, \ldots, r'\}$;
(22) **end**
(23) **end**
(24) **end**
(25) Retirar de $NDE(Y)$ os pontos que não são extremos;
(26) **end**

---

A linha (25) é necessária se estivermos interessados apenas nos pontos extremos de $Y$, pois algumas soluções não-dominadas não-extremas podem ser calculadas. Como $Y$ é um subconjunto de $\mathbb{R}^2$, a identificação dos pontos extremos que são soluções não-dominadas faz-se rapidamente a partir do conjunto $NDE(Y)$ obtido, depois de ordenado, por exemplo, por ordem crescente da primeira coordenada dos seus elementos.

### 5.3.2 Algoritmo primal-dual modificado

O algoritmo primal-dual, como foi apresentado na Subsecção 5.3.1, divide o intervalo ]0,1[ de acordo com os pontos de divisão da função linear por troços $\theta(\lambda)$. No entanto, nem todos estes subintervalos produzem uma solução eficiente diferente para (5.9) e pode acontecer que a mesma solução seja encontrada repetidamente. Nesta secção mostramos que esta redundância pode ser evitada. De facto, no caso bi-objectivo, podemos ir mais além e, possivelmente, descobrir um subintervalo com uma amplitude superior, onde a solução actual continua óptima, mais precisamente, dada uma solução eficiente, é possível determinar para que valores de $\lambda$ a próxima solução eficiente diferente será encontrada obtendo-se, assim, um conjunto de subintervalos $\Lambda_q, q = 1, 2, \ldots, r$ que constituem uma divisão do intervalo ]0,1[.

O algoritmo primal-dual divide o intervalo $]0, 1[$ sucessivamente de acordo com a árvore da Figura 5.8.

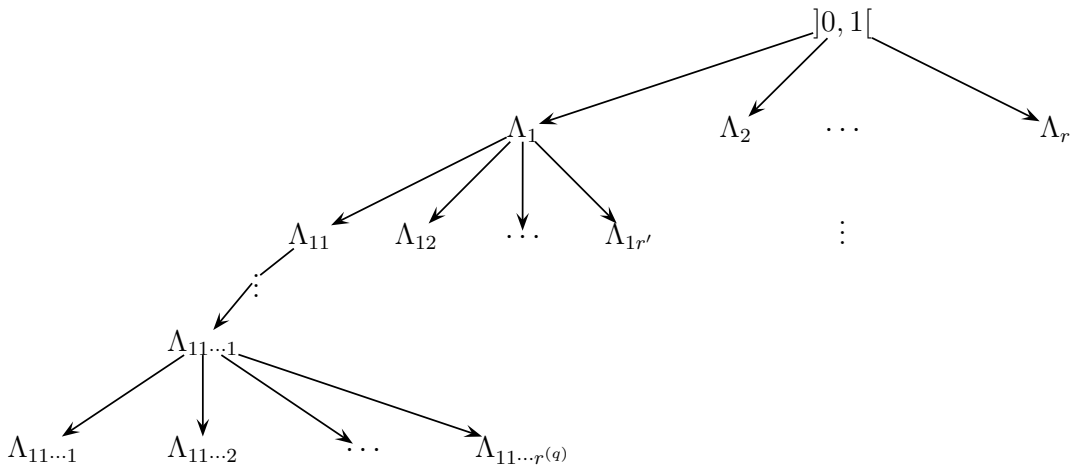



**Figura 5.8:** *Esquema de divisão do intervalo* $\Lambda$.

Seguindo um esquema de pesquisa em profundidade da esquerda para a direita, a primeira solução eficiente $x^*$ obtida (associada a $\Lambda_{11\cdots1}$) é uma solução óptima para o problema (5.10) com $\lambda = \varepsilon$, sendo $\varepsilon$ um número positivo suficientemente pequeno. Esta solução minimiza o segundo objectivo e a sua imagem no espaço dos objectivos é um ponto extremo de $Y$.

Consideremos os conjuntos $L$ e $U$ dos arcos no limite inferior e superior, respectivamente. Como vimos anteriormente $c_{ij}$ e $\pi_i$ são funções afins de $\lambda$. Assim, os

custos reduzidos são também funções afins de $\lambda$ e temos:

$$\begin{aligned}\bar{c}_{ij}(\lambda) = (c^1_{ij} - c^2_{ij})\lambda + c^2_{ij} - \pi_i(\lambda) + \pi_j(\lambda) &\geqslant 0,\ \forall (i,j) \in L,\\ \bar{c}_{ij}(\lambda) = (c^1_{ij} - c^2_{ij})\lambda + c^2_{ij} - \pi_i(\lambda) + \pi_j(\lambda) &\leqslant 0,\ \forall (i,j) \in U.\end{aligned}$$

Designemos por $\lambda' \in ]0,1[$ o valor de $\lambda'$:

$$\begin{aligned}\lambda' = \max\{ \quad & \lambda : \bar{c}_{ij}(\lambda) \geqslant 0,\ \forall (i,j) \in L \\ & \text{e } \bar{c}_{ij}(\lambda) \leqslant 0,\ \forall (i,j) \in U\}.\end{aligned} \tag{5.13}$$

A solução $x^*$ permanece óptima para o problema (5.10) para todo o $\lambda \in ]0, \lambda']$. Deste modo, na segunda iteração do algoritmo primal-dual, todos os subintervalos $\Lambda_q \subseteq ]0, \lambda']$ podem ser removidos de futuras análises uma vez que a mesma solução $x^*$ ou uma com o mesmo valor objectivo seria obtida nesses intervalos.

De forma semelhante, sempre que o algoritmo encontra uma nova solução eficiente os custos reduzidos dos arcos dos conjuntos $L$ e $U$ associados e o valor de $\lambda'$ são calculados como em (5.13). Todos os subintervalos para os quais sabemos que já existe uma solução (eficiente) são apagados de análises futuras. Verificar-se-á também se um subintervalo encontrado posteriormente no algoritmo verifica esta condição (ver Algoritmo 6).

No início da aplicação do algoritmo fazemos $\lambda^* = 0$. Na Linha (19) actualizamos o seu valor, onde $\lambda'$ é calculado de acordo com (5.13) e apagamos todos os $l$ de $\mathcal{L}$ para os quais $\Lambda_q \subseteq [0, \lambda^*]$.

*Algoritmo 6*: Algoritmo primal-dual bi-objectivo modificado.
{*Determina o conjunto das soluções não-dominadas extremas.* }

---

**input:** Rede orientada $\mathcal{N} = (\mathcal{G}, (c^1, c^2), l, u, b)$.
**output:** Conjunto de soluções não-dominadas extremas: $NDE(Y)$.

(1) **begin**
(2) $\Lambda_0 \leftarrow ]0,1[$, $\pi^0 \leftarrow 0$, $\lambda^* \leftarrow 0$ e $NDE(Y) \leftarrow \emptyset$;
(3) $\mathcal{L} \leftarrow \{(\Lambda_0, \mathcal{A}_0^= = \emptyset, L_0 = \mathcal{A}, U_0 = \emptyset, \pi^0 = 0, \mu^0 = 0)\}$;
(4) **while** $(\mathcal{L} \neq \emptyset)$ **do**
(5) **begin**
(6) $l \leftarrow (\Lambda_q, \mathcal{A}_q^=, L_q, U_q, \pi^q, \mu^q) \in \mathcal{L}$ com o menor valor de $\lambda_q$ ($\lambda_q$ e $\lambda_Q$
(7) são os extremos do intervalo $\Lambda_q$);
(8) **if** $(\lambda_Q \leqslant \lambda^*)$ **do** $\mathcal{L} = \mathcal{L} \setminus \{l\}$ e ir para (4); **else**
(9) **begin**
(10) $\hat{y} \leftarrow$ valor óptimo de $AP(\mathcal{A}_q^=)$;
(11) **if** $(\hat{y} = 0)$ **do**
(12) **begin**
(13) $\mathcal{L} \leftarrow \mathcal{L} \setminus \{l\}$, $x \leftarrow$ solução primal associada;
(14) $L \leftarrow$ arcos com fluxo igual ao limite inferior;
(15) $U \leftarrow$ arcos com fluxo igual ao limite superior;
(16) $\lambda' \leftarrow \max\{\lambda : \bar{c}_{ij}(\lambda) \geqslant 0 \ \forall (i,j) \in L$ e $\bar{c}_{ij}\ (\lambda) \leqslant 0 \ \forall (i,j) \in U\}$
(17) onde $\bar{c}_{ij}(\lambda) = (c_{ij}^1 - c_{ij}^2) \times \lambda + c_{ij}^2 - \pi_i(\lambda) + \pi_j(\lambda)$ é o custo
(18) reduzido do arco $(i,j)$;
(19) $\lambda^* \leftarrow \max\{\lambda', \lambda_Q\}$;
(20) $y^* \leftarrow$ imagem da solução do primal associada a $\hat{y} = 0$;
(21) $NDE(Y) \leftarrow NDE(Y) \cup \{y^*\}$;
(22) **end**
(23) **else**
(24) **begin**
(25) $(\hat{\pi}, \hat{\mu}) \leftarrow$ solução óptima de $DAP(\mathcal{A}_q^=)$;
(26) Calcular a divisão $\Lambda_{qq'}$, $q' = 1, \ldots, r'$ de $\Lambda_q$ de acordo com
(27) (5.12) e a nova solução dual $\pi^{qq'}(\lambda), \mu^{qq'}(\lambda)$ para todo o $q' =$
(28) $1, \ldots, r'$;
(29) $\mathcal{L} \leftarrow \mathcal{L} \setminus \{l\} \cup \{(\Lambda_{qq'}, L_{qq'}, U_{qq'}, \pi^{qq'}(\lambda), \mu^{qq'}(\lambda)) :$
(30) $q' = 1, \ldots, r'\}$;
(31) **end**
(32) **end**
(33) **end**
(34) Retirar de $NDE(Y)$ os pontos que não são extremos;
(35) **end**

---

### 5.3.3 Exemplo ilustrativo

Consideremos o problema de fluxos em redes bi-objectivo representado na Figura 5.9.

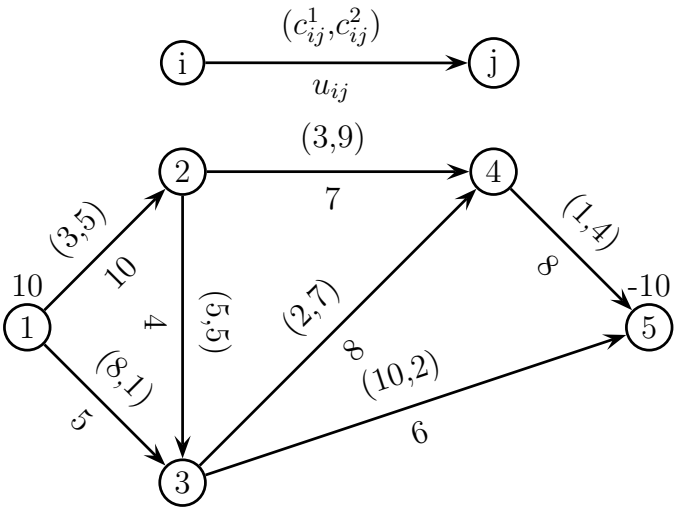



**Figura 5.9:** *Problema de fluxos em redes bi-objectivo.*

Matematicamente, este problema FRBO pode ser formulado da seguinte forma:

$$
\begin{aligned}
&\text{minimizar } f_1(x) = 3x_{12} + 8x_{13} + 5x_{23} + 3x_{24} + 2x_{34} + 10x_{35} + x_{45},\\
&\text{minimizar } f_2(x) = 5x_{12} + x_{13} + 5x_{23} + 9x_{24} + 7x_{34} + 2x_{35} + 4x_{45},\\
&\text{sujeito a:}\quad
\begin{array}{rrrrrrrcl}
x_{12} & +x_{13} & & & & & & = & 10,\\
-x_{12} & & +x_{23} & +x_{24} & & & & = & 0,\\
& -x_{13} & -x_{23} & & +x_{34} & +x_{35} & & = & 0,\\
& & & -x_{24} & -x_{34} & & +x_{45} & = & 0,\\
& & & & & -x_{35} & -x_{45} & = & -10,
\end{array}
\end{aligned}
\tag{5.14}
$$

$$
\begin{aligned}
&x_{12} \leqslant 10, x_{13} \leqslant 5, x_{23} \leqslant 4, x_{24} \leqslant 7, x_{34} \leqslant 8, x_{35} \leqslant 6,\\
&x_{45} \leqslant 8,\ x_{12}, x_{13}, x_{23}, x_{24}, x_{34}, x_{35}, x_{45} \geqslant 0.
\end{aligned}
\tag{5.15}
$$

Consideremos um parâmetro $\lambda \in ]0,1[$. O problema paramétrico associado é:

$$
\begin{array}{lll}
\text{minimizar} & (-2\lambda+5)x_{12}+(7\lambda+1)x_{13}+5x_{23}+(-6\lambda+9)x_{24}+ & \\
& +(-5\lambda+7)x_{34}+(8\lambda+2)x_{35}+(-3\lambda+4)x_{45}, & \\
\text{sujeito a:} & \begin{array}{rrrrrrcl} x_{12} & +x_{13} & & & & & = & 10, \\ -x_{12} & & +x_{23} & +x_{24} & & & = & 0, \\ & -x_{13} & -x_{23} & & +x_{34} & +x_{35} & = & 0, \\ & & & -x_{24} & -x_{34} & & +x_{45} = & 0, \\ & & & & & -x_{35} & -x_{45} = & -10, \end{array} & (5.16) \\
& x_{12} \leqslant 10, x_{13} \leqslant 5, x_{23} \leqslant 4, x_{24} \leqslant 7, x_{34} \leqslant 8, x_{35} \leqslant 6, & \\
& x_{45} \leqslant 8,\ x_{12}, x_{13}, x_{23}, x_{24}, x_{34}, x_{35}, x_{45} \geqslant 0. &
\end{array}
$$

Designemos por $\pi_1, \pi_2, \pi_3, \pi_4, \pi_5$ as variáveis duais associadas às restrições (5.14) e por $\mu_{12}, \mu_{13}, \cdots, \mu_{45}$ as variáveis duais associadas às restrições (5.15). Deste modo o dual do problema paramétrico pode ser definido como:

$$
\begin{array}{lll}
\text{maximizar} & 10\pi_1 - 10\pi_5 - 10\mu_{12} - 5\mu_{13} - 4\mu_{23} - 7\mu_{24} - 8\mu_{34} - 6\mu_{35} - 8\mu_{45}, & \\
\text{sujeito a:} & \begin{array}{rrrrrrcl} \pi_1 & -\pi_2 & & & & -\mu_{12} & \leqslant & -2\lambda+5, \\ \pi_1 & & -\pi_3 & & & -\mu_{13} & \leqslant & 7\lambda+1, \\ & \pi_2 & -\pi_3 & & & -\mu_{23} & \leqslant & 5, \\ & \pi_2 & & -\pi_4 & & -\mu_{24} & \leqslant & -6\lambda+9, \\ & & \pi_3 & -\pi_4 & & -\mu_{34} & \leqslant & -5\lambda+7, \\ & & \pi_3 & & -\pi_5 & -\mu_{35} & \leqslant & 8\lambda+2, \\ & & & \pi_4 & -\pi_5 & -\mu_{45} & \leqslant & -3\lambda+4, \end{array} & \\
& \mu_{12}, \mu_{13}, \mu_{23}, \mu_{24}, \mu_{34}, \mu_{35}, \mu_{45} \geqslant 0. &
\end{array}
$$

Apliquemos o Algoritmo 6 com o objectivo de encontrar todas as soluções não-dominadas extremas para o problema (5.14). Ao longo desta aplicação evidenciaremos as principais diferenças da aplicação deste algoritmo relativamente ao Algoritmo 5.

**$\Lambda_0$:** O algoritmo começa por considerar o problema paramétrico com um valor de $\lambda$ no intervalo $\Lambda_0 = ]0,1[$. Uma solução admissível para o dual deste problema tem todas as componentes nulas, isto é, podemos considerar $\pi^0 = 0$. O problema primal auxiliar ($AP(\mathcal{A}^=)$), com variáveis artificiais apenas, tem valor óptimo $\hat{z} = 10$, como se pode ver imediatamente na rede respectiva. No caso geral, podemos obter esta solução utilizando o Algoritmo 3. Como $\hat{z} \neq 0$ a solução obtida não é óptima para o problema paramétrico inicial. O problema dual auxiliar tem valor óptimo 10 com $\hat{\pi} = (0,1,1,1,-1)$. Estes valores são rapidamente obtidos utilizando a rede do problema auxiliar. O cálculo de

$\theta(\lambda)$ (ver Tabela 5.2) conduz-nos à decomposição de $\Lambda_0$ em dois subintervalos: $\Lambda_1 = ]0, \frac{2}{11}]$ e $\Lambda_2 = [\frac{2}{11}, 1[$, da seguinte forma:

$$\theta(\lambda) = \min\left\{4\lambda + 1,\ \frac{-3}{2}\lambda + 2\right\} = \begin{cases} 4\lambda + 1, & \lambda \in ]0, \frac{2}{11}] \\ \frac{-3}{2}\lambda + 2, & \lambda \in [\frac{2}{11}, 1[ \end{cases}$$

e

$$\begin{aligned} \pi^1(\lambda) &= \pi^0 + \theta(\lambda)\hat{\pi} \\ &= (0, 4\lambda+1, 4\lambda+1, 4\lambda+1, -4\lambda-1) \text{ se } \lambda \in ]0, \frac{2}{11}], \\ \pi^2(\lambda) &= \pi^0 + \theta(\lambda)\hat{\pi} \\ &= \left(0, \frac{-3}{2}\lambda + 2, \frac{-3}{2}\lambda + 2, \frac{-3}{2}\lambda + 2, \frac{3}{2}\lambda - 2\right) \text{ se } \lambda \in \left[\frac{2}{11}, 1\right[. \end{aligned}$$

**Tabela 5.2:** *Cálculos auxiliares para $\theta$ em $\Lambda_0$.*

| arc | $c_{ij} - \pi_i^0 + \pi_j^0$ | $\hat{\pi}_i - \hat{\pi}_j$ | $\frac{c_{ij} - \pi_i^0 + \pi_j^0}{\hat{\pi}_i - \hat{\pi}_j}$ |
|---|---|---|---|
| $(1,2)$ | $-2\lambda + 5$ | $-1$ | |
| $(1,3)$ | $7\lambda + 1$ | $-1$ | |
| $(2,3)$ | $5$ | $0$ | |
| $(2,4)$ | $-6\lambda + 9$ | $0$ | |
| $(3,4)$ | $-5\lambda + 7$ | $0$ | |
| $(3,5)$ | $8\lambda + 2$ | $2$ | $\frac{8\lambda+2}{2} = 4\lambda + 1$ |
| $(4,5)$ | $-3\lambda + 4$ | $2$ | $\frac{-3\lambda+4}{2} = \frac{-3}{2}\lambda + 2$ |

Consideremos o primeiro subintervalo da divisão e passemos à segunda iteração.

**$\Lambda_1$:** Com $\lambda \in ]0, \frac{2}{11}]$ e a solução dual $\pi^1$ determinemos os novos conjuntos $\mathcal{A}^=$, $L_1$ e $U_1$. Temos $c_{ij} - \pi_i^1 + \pi_j^1 > 0$ para todo o arco $(i,j) \in \mathcal{A}$ excepto para $(3,5)$ que verifica $c_{35} - \pi_3^1 + \pi_5^1 = 0$. Isto implica que no novo problema auxiliar $x_{ij} = 0$, para todo o $(i,j) \in \mathcal{A}$ excepto $x_{35}$ (ver Tabela 5.3), isto é, $\mathcal{A}^= = \{(3,5)\}$, $L_1 = \mathcal{A} \setminus \{(3,5)\}$ e $U_1 = \emptyset$. O problema auxiliar tem valor óptimo $10 \neq 0$ e o novo dual auxiliar tem solução óptima $\pi = (0, 1, -1, 1, -1)$. Temos

$$\theta(\lambda) = \min\left\{11\lambda + 2, \frac{5}{2}, \frac{-11}{2}\lambda + 1\right\} = \frac{-11}{2}\lambda + 1,\ \forall\lambda \in \left]0, \frac{2}{11}\right[,$$

isto é, $\Lambda_1$ mantém-se sem qualquer decomposição. Actualizamos os valores da solução dual:

$$\pi^1(\lambda) = \left(0, -\frac{3}{2}\lambda + 2, \frac{19}{2}\lambda, -\frac{3}{2}\lambda + 2, \frac{3}{2}\lambda - 2\right).$$

**Tabela 5.3:** *Cálculos auxiliares para $\theta$ em $\Lambda_1$.*

| arc | $c_{ij} - \pi_i^1 + \pi_j^1$ | $\hat{\pi}_i - \hat{\pi}_j$ | $\frac{c_{ij}-\pi_i^1+\pi_j^1}{\hat{\pi}_i-\hat{\pi}_j}$ |
|---|---|---|---|
| $(1,2)$ | $2\lambda + 6$ | $-1$ | |
| $(1,3)$ | $11\lambda + 2$ | $1$ | $11\lambda + 2$ |
| $(2,3)$ | $5$ | $2$ | $\frac{5}{2}$ |
| $(2,4)$ | $-6\lambda + 9$ | $0$ | |
| $(3,4)$ | $-5\lambda + 7$ | $-2$ | |
| $(3,5)$ | $0$ | $0$ | |
| $(4,5)$ | $-11\lambda + 2$ | $2$ | $\frac{-11\lambda+2}{2} = \frac{-11}{2}\lambda + 1$ |

O novo problema auxiliar tem $x_{12} = x_{13} = x_{23} = x_{24} = x_{34} = 0$ com valor óptimo $10 \neq 0$. A solução óptima do problema dual associada é $\hat{\pi} = (0, 1, -1, -1, -1)$. Assim,

$$\theta(\lambda) = \min\left\{\frac{33}{2}\lambda + 1, \frac{11\lambda + 3}{2}, -3\lambda + \frac{9}{2}\right\} = \left\{\begin{array}{ll} \frac{33}{2}\lambda + 1, & \lambda \in ]0, \frac{1}{22}] \\ \frac{11\lambda+3}{2}, & \lambda \in [\frac{1}{22}, \frac{2}{11}[ \end{array}\right. ,$$

(ver Tabela 5.4) e $\Lambda_1$ é decomposto em $\Lambda_{11} = ]0, \frac{1}{22}]$ e $\Lambda_{12} = [\frac{1}{22}, \frac{2}{11}]$, com

$$\begin{aligned} \pi^{11}(\lambda) &= (0, 15\lambda + 3, -7\lambda - 1, -18\lambda + 1, -15\lambda - 3), \forall\lambda \in \left]0, \frac{1}{22}\right], \\ \pi^{12}(\lambda) &= \left(0, \frac{5}{4}\lambda + \frac{7}{2}, \frac{27}{4}\lambda - \frac{3}{2}, -\frac{17}{4}\lambda + \frac{1}{2}, -\frac{5}{4}\lambda - \frac{7}{2}\right), \forall\lambda \in \left[\frac{1}{22}, \frac{2}{11}\right]. \end{aligned}$$

Consideremos o intervalo $\Lambda_{11}$ e passemos à iteração seguinte.

**Tabela 5.4:** *Cálculos auxiliares para $\theta$ em $\Lambda_1$ ($2^o$ caso).*

| arc | $c_{ij} - \pi_i^1 + \pi_j^1$ | $\hat{\pi}_i - \hat{\pi}_j$ | $\frac{c_{ij}-\pi_i^1+\pi_j^1}{\hat{\pi}_i-\hat{\pi}_j}$ |
|---|---|---|---|
| $(1,2)$ | $-\frac{7}{2}\lambda + 7$ | $-1$ | |
| $(1,3)$ | $\frac{33}{2}\lambda + 1$ | $1$ | $\frac{33}{2}\lambda + 1$ |
| $(2,3)$ | $11\lambda + 3$ | $2$ | $\frac{11\lambda+3}{2}$ |
| $(2,4)$ | $-6\lambda + 9$ | $2$ | $\frac{-6\lambda+9}{2} = -3\lambda + \frac{9}{2}$ |
| $(3,4)$ | $-16\lambda + 9$ | $0$ | |
| $(3,5)$ | $0$ | $0$ | |
| $(4,5)$ | $0$ | $0$ | |

**$\Lambda_{11}$:** O problema auxiliar tem $x_{12} = x_{23} = x_{24} = x_{34} = 0$ e o seu valor óptimo é $5 \neq 0$. O problema dual auxiliar tem solução óptima $\hat{\pi} = (0, 1, -1, -1, -1)$ e

$$\theta(\lambda) = \min\left\{-11\lambda + \frac{1}{2}, \frac{-39\lambda + 7}{2}\right\} = -11\lambda + \frac{1}{2},$$

$\forall \lambda \in \Lambda_{11}$ e $\pi^{11}(\lambda)$ é actualizada para,

$$\pi^{11}(\lambda) = \left(0, 4\lambda + \frac{7}{2}, 4\lambda - \frac{3}{2}, -7\lambda + \frac{1}{2}, -4\lambda - \frac{7}{2}\right).$$

Os custos reduzidos dos arcos respectivos implicam que $x_{12} = x_{24} = x_{34} = 0$ e que $x_{13} = u_{13} = 5$. O problema primal auxiliar tem valor óptimo $5 \neq 0$ e o problema dual correspondente tem solução óptima $\hat{\pi} = (0, -1, -1, -1, -1)$, onde

$$\theta(\lambda) = \min\left\{2\lambda + \frac{17}{2}\right\} = 2\lambda + \frac{17}{2}.$$

A solução dual admissível é actualizada, para $\lambda \in ]0, \frac{1}{22}]$, da seguinte maneira:

$$\pi^{11}(\lambda) = (0, 2\lambda - 5, 2\lambda - 10, -9\lambda - 8, -6\lambda - 12).$$

O problema primal auxiliar seguinte tem $x_{13} = 5$, $x_{24} = x_{34} = 0$ e valor óptimo $4 \neq 0$. O problema dual associado tem solução óptima $\hat{\pi} = (0, 0, 0, -1, -1)$. Temos assim:

$$\begin{aligned} \theta(\lambda) &= \min\{-17\lambda + 6, 16\lambda + 9\} = -17\lambda + 6, \\ \pi^{11}(\lambda) &= (0, 2\lambda - 5, 2\lambda - 10, 8\lambda - 14, 11\lambda - 18). \end{aligned}$$

O valor dos custos reduzidos implica que o próximo problema auxiliar tenha $x_{13} = 5$, $x_{34} = 0$ e $x_{35} = u_{35} = 6$. Este problema tem valor óptimo 0 e, por isso, encontramos uma solução que é óptima para o problema inicial (5.16) para $\lambda \in [0, \frac{1}{22}]$. Essa solução é $x_{12} = 5$, $x_{13} = 5$, $x_{23} = 1$, $x_{24} = 4$, $x_{34} = 0$, $x_{35} = 6$ e $x_{45} = 4$ com $f_1(x) = 136$ e $f_2(x) = 99$. Na versão original do algoritmo primal-dual, continuávamos à procura de uma nova solução no intervalo $\Lambda_{12} = [\frac{1}{22}, \frac{2}{11}]$. No algoritmo modificado, propomos o uso da informação contida na solução obtida de forma a reduzir trabalho posterior. Esta solução tem arcos $(3,4) \in L$ e $(1,3),(3,5) \in U$ e é uma solução óptima do problema paramétrico para $\lambda \in ]0, \frac{6}{17}]$ tendo em conta que $\bar{c}_{34} = \lambda + 3 \geqslant 0$ para todo o $\lambda \in ]0,1[$, $\bar{c}_{13} = 9\lambda - 9 \leqslant 0$ para todo o $\lambda \in ]0,1[$ e $\bar{c}_{35} = 17\lambda - 6 \leqslant 0$ para $\lambda \in ]0, \frac{6}{17}]$. Como $[\frac{1}{22}, \frac{2}{11}] \subset \lambda \in ]0, \frac{6}{17}]$ o algoritmo continua com $\lambda \in [\frac{2}{11}, 1]$.

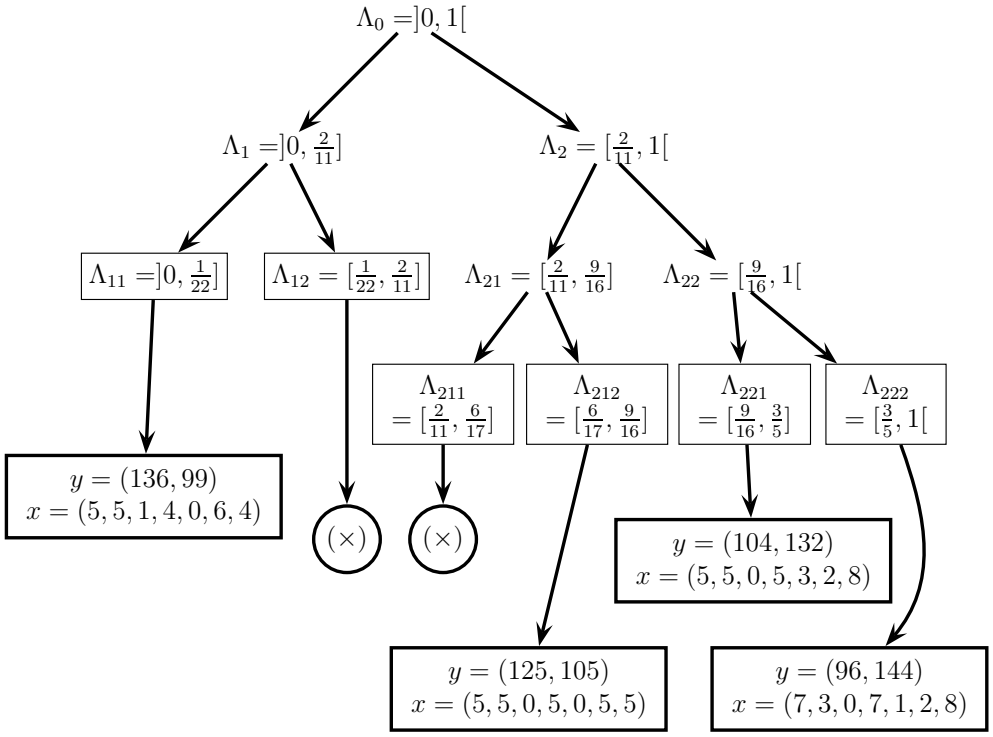



**Figura 5.10:** *Esquema com subintervalos e soluções.*

**$\Lambda_2$:** O uso do algoritmo para $\lambda \in \Lambda_2$ conduz à subárvore do subintervalo $\Lambda_2$ na Figura 5.10. O subintervalo $\Delta_{211}$ pode ser descartado uma vez que sabemos

que a mesma solução não-dominada seria obtida neste intervalo. Ao explorar os subintervalos $\Delta_{212}$, $\Delta_{221}$ e $\Delta_{222}$, as soluções não-dominadas $(125, 105)$, $(104, 132)$ e $(96, 144)$ e as soluções eficientes correspondentes $(5, 5, 0, 5, 0, 5, 5)$, (5,5,0, 5, 3, 2, 8) e $(7, 3, 0, 7, 1, 2, 8)$ são obtidas.

O algoritmo termina; a forma como o subintervalo $]0, 1[$ foi subdividido e as soluções encontradas são apresentadas na Figura 5.10. O símbolo $(\times)$ indica que o intervalo foi apagado devido aos cálculos de $\lambda^*$ depois de encontrada uma solução.

## 5.4 Experiências computacionais, resultados e comentários

Foram implementados dois algoritmos, para o problema de fluxos em redes, usando linguagem de programação `C`:

**(pd)** O Algoritmo 6 e

**(par)** um algoritmo paramétrico como descrito na Subsecção 5.2.1, usando o simplex para redes para determinar a primeira solução eficiente. O algoritmo paramétrico implementado calcula todas as soluções óptimas alternativas de forma a assegurar que todos os pontos eficientes extremos são encontrados (ver página 122 em Steuer, 1986).

O computador usado para os testes estava equipado com um processador Intel Pentium 2.13GHz e 1GB de memória RAM e foi utilizado o sistema operativo OS X.

**Tabela 5.5:** *Parâmetros do problema ($u_{ij}$ designa a capacidade, $m$ designa o número de nodos e $n$ o número de arcos).*

| Tipo de problema | $m$ | $n$ | Origens | Destinos | Oferta Total | % Arcos da estrutura com $c_{ij}^1$ max | com $c_{ij}^2$ max | $u_{ij}$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 10 | 40 | 5 | 4 | 100 | 20 | 20 | 0 – 20 |
| 2 | 20 | 100 | 7 | 5 | 200 | 30 | 30 | 0 – 30 |
| 3 | 30 | 300 | 8 | 12 | 300 | 25 | 25 | 0 – 30 |
| 4 | 40 | 600 | 12 | 14 | 400 | 20 | 30 | 0 – 40 |
| 5 | 50 | 1000 | 15 | 15 | 600 | 25 | 25 | 0 – 40 |
| 6 | 60 | 1400 | 20 | 20 | 800 | 20 | 20 | 0 – 40 |

**Tabela 5.6:** *Resultados numéricos para todas as instâncias.*

| Tipo de instância | Algoritmo | Média de NDE(Y) | Tempo *CPU* Max | Média | Mediana | Min | Moda |
|---|---|---|---|---|---|---|---|
| 1 | pd | 7.77 | 0.03 | 0.01 | 0.02 | 0 | 0 |
| | par | | 0.06 | 0.02 | 0 | 0 | 0 |
| 2 | pd | 29.63 | 0.20 | 0.13 | 0.14 | 0.08 | 0.14 |
| | par | | 0.08 | 0.03 | 0.03 | 0.02 | 0.02 |
| 3 | pd | 85.67 | 3.16 | 2.32 | 2.38 | 1.45 | 1.94 |
| | par | | 0.11 | 0.06 | 0.05 | 0.03 | 0.05 |
| 4 | pd | 126.77 | 14.38 | 10.30 | 10.29 | 5.47 | 5.47 |
| | par | | 0.27 | 0.19 | 0.19 | 0.14 | 0.19 |
| 5 | pd | 208.33 | 75.05 | 50.74 | 50.28 | 32.73 | 32.73 |
| | par | | 0.98 | 0.68 | 0.66 | 0.48 | 0.59 |
| 6 | pd | 281.03 | 267.89 | 138.18 | 133.13 | 68.13 | 68.13 |
| | par | | 1.80 | 1.23 | 1.19 | 0.89 | 1.11 |

Um conjunto de 180 instâncias do problema de fluxos em redes foi gerado, usando o gerador de redes *NETGEN* adaptado a este problema particular. Foram considerados seis tipos de problemas com 30 instâncias para cada um deles. Os coeficientes das funções objectivo foram gerados aleatoriamente (usando a distribuição uniforme) do conjunto de inteiros $\{0, 1, 2, \cdots, 100\}$. Todos os arcos têm uma capacidade limitada. Os restantes parâmetros dos problemas gerados estão listados na Tabela 5.5 seguindo as convenções do *NETGEN* (ver Klingman *et al.*, 1974).

**Tabela 5.7:** *Resultados numéricos para as instâncias altamente degeneradas.*

| Nodos | Arcos | NDE(Y) | Bases eficientes | Tempo *CPU* pd | par |
|---|---|---|---|---|---|
| 10 | 42 | 7 | 37 | 0.00 | 0.22 |
| 10 | 42 | 8 | 35 | 0.00 | 1.25 |
| 20 | 100 | 23 | 2 333 | 0.20 | 1.41 |
| 20 | 100 | 21 | 2 692 | 0.23 | 2.66 |
| 20 | 100 | 16 | 2 998 | 0.14 | 1.49 |
| 20 | 100 | 13 | 7 431 | 0.11 | 8.55 |
| 20 | 100 | 18 | 2 041 | 0.20 | 1.27 |
| 20 | 100 | 20 | 3 259 | 0.17 | 2.64 |
| 20 | 100 | 13 | 2 979 | 0.16 | 3.80 |
| 20 | 100 | 22 | 5 782 | 0.30 | 4.83 |

*continua na próxima página*

*continuação da página anterior*

| Nodos | Arcos | NDE(Y) | Bases eficientes | Tempo pd | *CPU* par |
|---|---|---|---|---|---|
| 20 | 100 | 18 | 2 581 | 0.24 | 1.53 |
| 20 | 100 | 11 | 3 219 | 0.08 | 1.56 |
| 20 | 100 | 17 | 2 802 | 0.17 | 1.22 |
| 20 | 100 | 17 | 4 372 | 0.19 | 2.38 |
| 30 | 300 | 55 | 6 007 | 1.52 | 10.70 |
| 30 | 300 | 73 | 1 322 | 1.92 | 2.41 |
| 30 | 300 | 56 | 8 096 | 1.50 | 17.20 |
| 30 | 300 | 59 | 2 134 | 1.91 | 1.58 |
| 30 | 300 | 54 | 9 493 | 1.25 | 7.63 |
| 30 | 300 | 53 | 13 528 | 1.48 | 16.99 |
| 30 | 300 | 51 | 5 541 | 1.44 | 5.53 |
| 30 | 300 | 49 | 8 902 | 1.69 | 7.81 |
| 30 | 300 | 56 | 17 433 | 1.63 | 23.34 |
| 40 | 600 | 68 | 4 500 | 5.19 | 7.56 |
| 40 | 600 | 49 | 17 912 | 2.36 | 38.33 |
| 40 | 600 | 77 | 8 847 | 2.30 | 11.72 |
| 40 | 600 | 96 | 9 923 | 4.25 | 14.64 |
| 40 | 600 | 69 | 11 168 | 2.48 | 37.11 |
| 40 | 600 | 71 | 26 659 | 2.91 | 178.20 |
| 40 | 600 | 71 | 16 631 | 2.25 | 37.64 |
| 40 | 600 | 78 | 16 291 | 2.59 | 37.39 |

O algoritmo paramétrico gasta um tempo menor que o primal-dual para a maioria das instâncias (ver Tabela 5.6). No entanto, para algumas instâncias o algoritmo primal-dual é mais rápido (ver Tabela 5.7 onde são apresentados alguns exemplos). Os custos de 31 instâncias da Tabela 5.7 foram gerados de forma a que as instâncias resultantes fossem altamente degeneradas. Todas estas instâncias têm um grande número de bases eficientes, muito maior que o número de pontos não-dominados extremos, significando isto que existem muitas soluções degeneradas. Os nossos testes computacionais parecem apontar que o algoritmo primal-dual é mais adequado para resolver este tipo de instâncias difíceis. A Figura 5.11 mostra que, quando o número de bases eficientes cresce quando comparado com o número de soluções não-dominadas, o tempo de *CPU* utilizado pelo algoritmo primal-dual decresce quando comparado com este mesmo tempo gasto pelo algoritmo paramétrico. Neste gráfico $u = \dfrac{\text{número de bases eficientes}}{\text{número de soluções não-dominadas extremas}}$ e $v = \dfrac{\text{tempo } CPU \text{ gasto por pd}}{\text{tempo } CPU \text{ gasto por par}}$. 211 instâncias foram consideradas, as 180 instâncias da Tabela 5.5 mais 31 instâncias adicionais geradas.

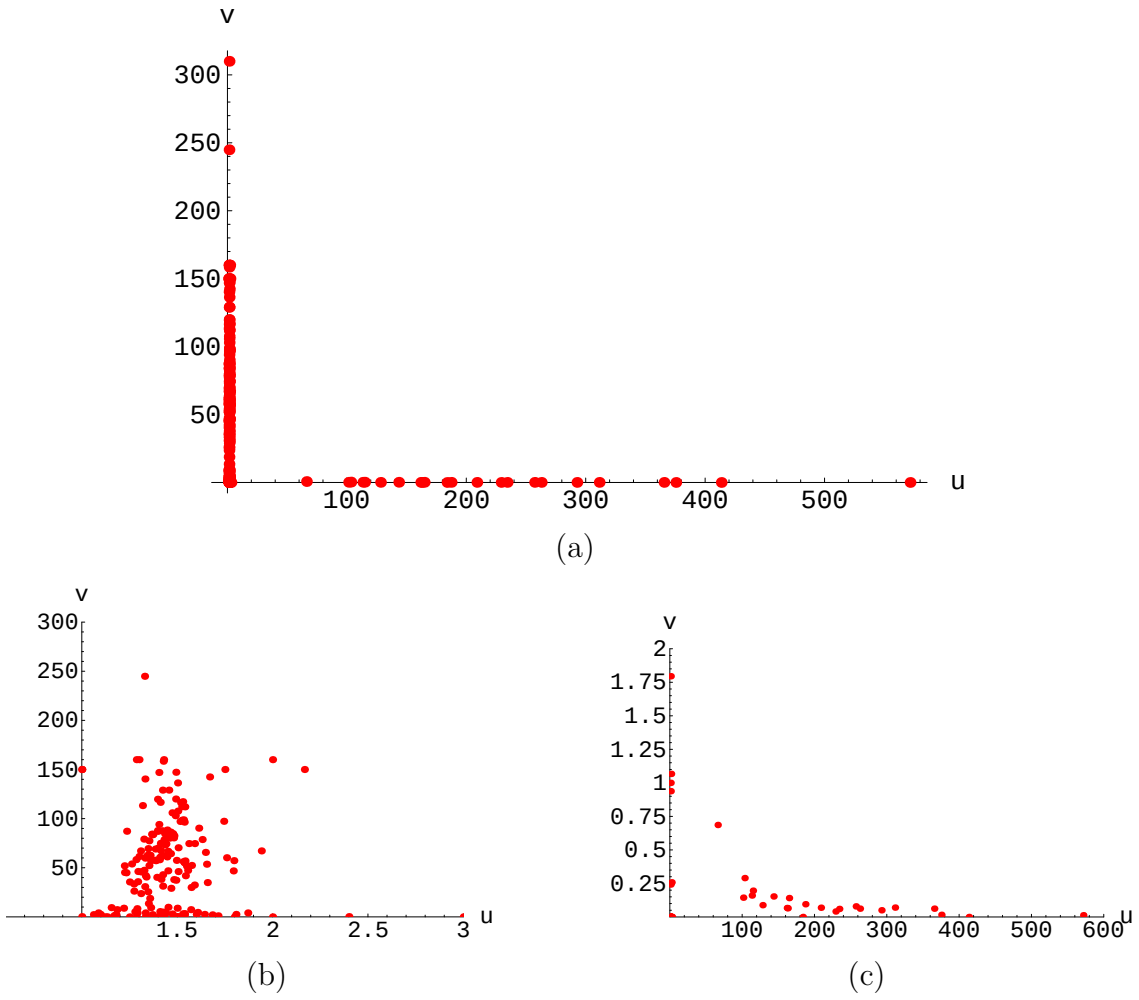



**Figura 5.11:** *(a) Número de bases eficientes por pontos não-dominados extremos (u) versus taxa de tempo CPU gasto pelo algoritmo pd sobre o tempo CPU gasto pelo algoritmo par (v); (b)* $0 \leqslant u \leqslant 3$*;(c)* $0 \leqslant v \leqslant 2$*.*

# Parte III

# Problemas lineares de fluxos em redes inteiros multi-objectivo

Nesta parte apresentamos o trabalho realizado no âmbito dos problemas dos fluxos inteiros em redes com vários objectivos. No Capítulo 6, descrevemos um novo algoritmo para identificar todas as soluções não-dominadas ou eficientes suportadas para este problema. O algoritmo baseia-se nas condições de optimalidade de uma solução cuja rede residual associada não contém qualquer circuito de custo negativo e, no caso de haver soluções óptimas alternativas, estas estão associadas com circuitos de custo nulo. Assim, utilizando uma função objectivo adequada, podemos determinar o conjunto das soluções não-dominadas ou eficientes suportadas, procurando os circuitos de custo nulo, nas redes residuais associadas às soluções óptimas que vão sendo determinadas. Todo o algoritmo é construído com base na demonstração de um teorema que mostra a conexidade do conjunto de todas as soluções não-dominadas suportadas com base no conceito de ciclo de custo zero. O algoritmo proposto determina apenas as soluções não-dominadas suportadas, embora se pense ser possível generalizá-lo para determinar o conjunto de todas as soluções não-dominadas. O algoritmo foi implementado e testado e os resultados fazem parte deste capítulo também. No Capítulo 7, descrevemos um algoritmo capaz de calcular todas as soluções não-dominadas e eficientes suportadas e não-suportadas para o problema bi-objectivo. Como é sabido as soluções eficientes/não-dominadas deste problema podem ser obtidas recorrendo à resolução de problemas de restrição-$\varepsilon$, com uma escolha adequada do valor $\varepsilon$. É essa a estratégia utilizada: através da escolha adequada do valor de $\varepsilon$, determinamos a solução óptima do problema de restrição-$\varepsilon$ associado, que nos dá uma das coordenadas da solução não-dominada correspondente. Queremos realçar aqui a forma como é feita a resolução do problema de restrição-$\varepsilon$. É sabido que a introdução da restrição-$\varepsilon$ no problema de fluxos em redes origina um novo problema que não é mais de fluxos em redes. No entanto, este problema pode ser resolvido utilizando apenas problemas de fluxos em redes, através de um procedimento de separação e avaliação progressivas, permitindo dessa forma tirar vantagem das propriedades deste tipo de problemas. O algoritmo proposto foi implementado e os resultados fazem parte deste capítulo, assim como a sua análise. Finalmente, no Capítulo 8, um novo algoritmo para o cálculo aproximado das soluções não--dominadas do problema de fluxos em redes é descrito. Este algoritmo utiliza a capacidade que o algoritmo proposto no Capítulo 7 tem de ir buscar qualquer solução eficiente/não-dominada. Deste modo, é possível obter uma representação do conjunto de todas as soluções eficientes/não-dominadas com determinadas características definidas, por exemplo, pelo decisor. Esta característica é de extrema importância considerando que não existe uma "boa" representação, do conjunto de todas as soluções eficientes/não-dominadas, mas existem alternativas diferentes, como veremos também neste capítulo.

# Capítulo 6

# Determinação de todas as soluções inteiras suportadas

Este capítulo é dedicado à determinação das soluções suportadas, num problema FRIMO. Começamos por fazer uma breve introdução ao capítulo, explicando como apareceu e a importância do algoritmo proposto. Seguidamente, fazemos uma caracterização das soluções não-dominadas como imagens das soluções eficientes. Mostramos, através de exemplos, que nem todas as soluções no conjunto das não-dominadas suportadas são imagens de soluções eficientes extremas ou intermédias. Depois, introduzimos conceitos teóricos, alguns dos quais são novos e descrevemos os diversos passos do algoritmo. Este algoritmo resulta da prova de que o conjunto das soluções suportadas é conexo. Apresentamos um exemplo ilustrativo da aplicação do algoritmo. Finalmente, descrevemos como foi feita a implementação do algoritmo, listamos os resultados obtidos da resolução de um conjunto de problemas e analisamos esses resultados.

## 6.1 Introdução

Este capítulo apresenta um novo algoritmo para determinar todas as soluções eficientes suportadas e todas as soluções não-dominadas suportadas, para o problema de fluxos em redes inteiro multi-objectivo. O algoritmo utiliza a conexidade do conjunto de soluções eficientes associadas com os circuitos de custo zero numa rede residual, demonstrada no Teorema 6.3.1. Convém referir que, tanto quanto é do nosso conhecimento, não havia até agora qualquer algoritmo dedicado ao cálculo das soluções não-dominadas suportadas, que conseguisse calcular todo o conjunto destas soluções. Um dos métodos mais utilizados, que foi usado por diversos investigadores como determinando todas estas soluções, verificou-se estar incorrecto, como mostramos na Secção 6.2.

## 6.2 Caracterização das soluções não-dominadas

Consideremos um problema FRIMO com regiões admissíveis $X^I$ e $Y^I$, no espaço das variáveis de decisão e no espaço dos objectivos, respectivamente, como definido na Subsecção 2.7.1. Contrariamente ao que acontece no problema FRMO, em que todas as soluções não-dominadas pertencem à fronteira da região admissível, neste problema podemos ter soluções não-dominadas na fronteira ou no interior de $conv(Y^I)$, isto é, podemos ter soluções não-dominadas suportadas ou não-suportadas, como vimos na Subsecção 2.7.2.

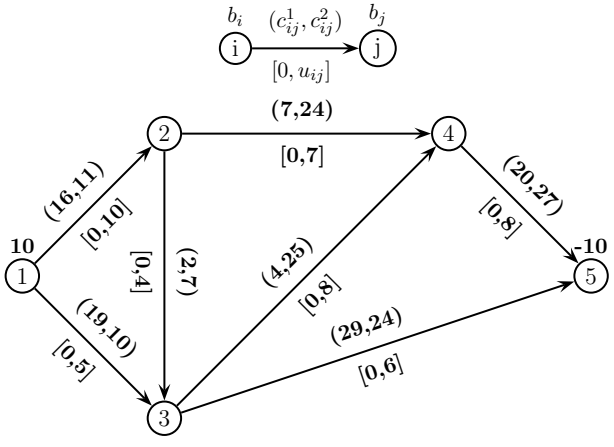



**Figura 6.1:** *Problema FRIBO.*

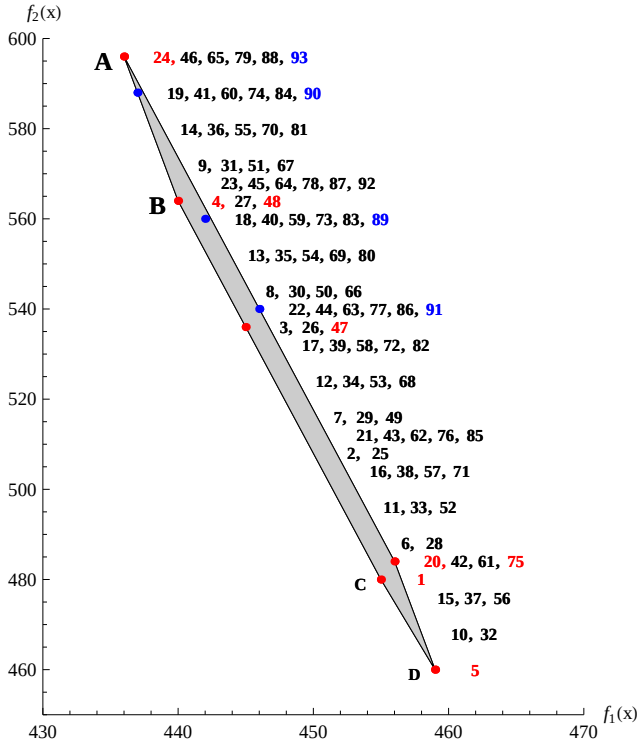



**Figura 6.2:** *Região admissível no espaço dos objectivos.*

Sabemos que todos os pontos extremos do poliedro $conv(Y^I)$ são imagens de pontos extremos do poliedro $conv(X^I)$, isto é, dado um ponto extremo $y \in conv(Y^I)$ existe sempre um ponto extremo $x \in conv(X^I)$ tal que $y = f(x)$ (ver Steuer, 1986, Zeleny, 1974). Além disso, como estamos a admitir que todas as componentes dos vectores $u$, $b$ e da matriz $C$ são inteiras, as soluções associadas aos pontos extremos têm componentes inteiras (ver Bazaraa *et al.*, 2005). Como é fácil de ver, nem todas as imagens de pontos extremos em $conv(X^I)$ são pontos extremos em $conv(Y^I)$. Como exemplo consideremos o problema FRIBO representado pela Figura 6.1. O conjunto das soluções admissíveis, no espaço das variáveis de decisão, tem 93 elementos no total, como pode ser visto em Figueira (2000) e na Tabela 6.1. A representação da região admissível no espaço dos objectivos está na Figura 6.2, com a numeração das soluções cujas imagens são representadas pelos pontos respectivos. O poliedro $conv(X^I)$ tem 12 pontos extremos correspondentes a 12 soluções básicas admissíveis, numeradas $1, 4, 5, 20, 24, 47, 48, 75, 89, 90, 91, 93$. Destas, as soluções 89, 90, 91 e 93 são degeneradas. Ao olhar para a Figura 6.2 podemos constatar que todos os pontos extremos do poliedro $conv(Y^I)$ são imagens de pelo

menos um ponto extremo de $conv(X^I)$. Alguns são imagens de apenas um ponto extremo, como é o caso do ponto 5, mas outros são imagens de mais que um ponto extremo como, por exemplo, o ponto extremo mais à esquerda que é imagem dos pontos extremos 24 e 93.

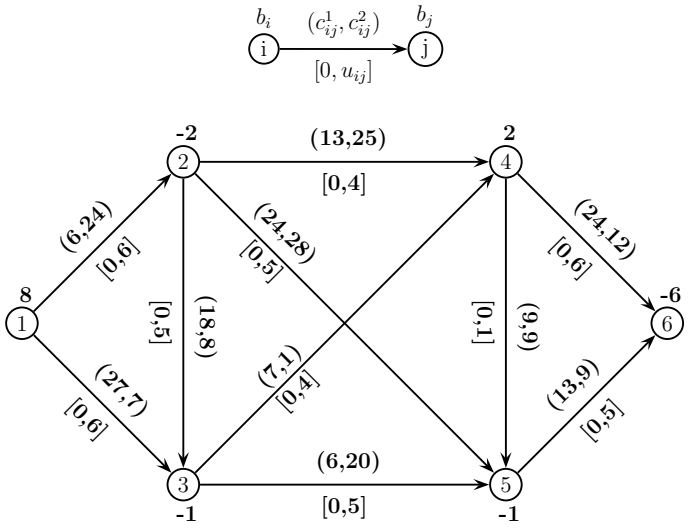



**Figura 6.3:** *Problema FRIBO.*

**Tabela 6.1:** *Soluções admissíveis do problema da Figura 6.1.*

| Nº | $x_{12}$ | $x_{13}$ | $x_{23}$ | $x_{24}$ | $x_{34}$ | $x_{35}$ | $x_{45}$ | $y_1$ | $y_2$ |
|---|---|---|---|---|---|---|---|---|---|
| **1** | **5** | **5** | **0** | **5** | **0** | **5** | **5** | **455** | **480** |
| 2 | 5 | 5 | 0 | 5 | 1 | 4 | 6 | 450 | 508 |
| 3 | 5 | 5 | 0 | 5 | 2 | 3 | 7 | 445 | 536 |
| **4** | **5** | **5** | **0** | **5** | **3** | **2** | **8** | **440** | **564** |
| **5** | **5** | **5** | **1** | **4** | **0** | **6** | **4** | **459** | **460** |
| 6 | 5 | 5 | 1 | 4 | 1 | 5 | 5 | 454 | 488 |
| 7 | 5 | 5 | 1 | 4 | 2 | 4 | 6 | 449 | 516 |
| 8 | 5 | 5 | 1 | 4 | 3 | 3 | 7 | 444 | 544 |
| 9 | 5 | 5 | 1 | 4 | 4 | 2 | 8 | 439 | 572 |
| 10 | 5 | 5 | 2 | 3 | 1 | 6 | 4 | 458 | 468 |
| 11 | 5 | 5 | 2 | 3 | 2 | 5 | 5 | 453 | 496 |
| 12 | 5 | 5 | 2 | 3 | 3 | 4 | 6 | 448 | 524 |
| 13 | 5 | 5 | 2 | 3 | 4 | 3 | 7 | 443 | 552 |

*continua na próxima página*

*continuação da página anterior*

| Nº | $x_{12}$ | $x_{13}$ | $x_{23}$ | $x_{24}$ | $x_{34}$ | $x_{35}$ | $x_{45}$ | $y_1$ | $y_2$ |
|---|---|---|---|---|---|---|---|---|---|
| 14 | 5 | 5 | 2 | 3 | 5 | 2 | 8 | 438 | 580 |
| 15 | 5 | **5** | 3 | 2 | **2** | **6** | 4 | 457 | 476 |
| 16 | 5 | 5 | 3 | 2 | 3 | 5 | 5 | 452 | 504 |
| 17 | 5 | 5 | 3 | 2 | 4 | 4 | 6 | 447 | 532 |
| 18 | 5 | 5 | 3 | 2 | 5 | 3 | 7 | 442 | 560 |
| 19 | 5 | 5 | 3 | 2 | 6 | 2 | 8 | 437 | 588 |
| **20** | **5** | **5** | **4** | **1** | **3** | **6** | **4** | **456** | **484** |
| 21 | 5 | 5 | 4 | 1 | 4 | 5 | 5 | 451 | 512 |
| 22 | 5 | 5 | 4 | 1 | 5 | 4 | 6 | 446 | 540 |
| 23 | 5 | 5 | 4 | 1 | 6 | 3 | 7 | 441 | 568 |
| **24** | **5** | **5** | **4** | **1** | **7** | **2** | **8** | **436** | **596** |
| 25 | 6 | 4 | 0 | 6 | 0 | 4 | 6 | 450 | 508 |
| 26 | 6 | 4 | 0 | 6 | 1 | 3 | 7 | 445 | 536 |
| 27 | 6 | 4 | 0 | 6 | 2 | 2 | 8 | 440 | 564 |
| 28 | 6 | 4 | 1 | 5 | 0 | 5 | 5 | 454 | 488 |
| 29 | 6 | 4 | 1 | 5 | 1 | 4 | 6 | 449 | 516 |
| 30 | 6 | **4** | 1 | **5** | **2** | **3** | 7 | 444 | 544 |
| 31 | 6 | 4 | 1 | 5 | 3 | 2 | 8 | 439 | 572 |
| 32 | 6 | 4 | 2 | 4 | 0 | 6 | 4 | 458 | 468 |
| 33 | 6 | 4 | 2 | 4 | 1 | 5 | 5 | 453 | 496 |
| 34 | 6 | 4 | 2 | 4 | 2 | 4 | 6 | 448 | 524 |
| 35 | 6 | 4 | 2 | 4 | 3 | 3 | 7 | 443 | 552 |
| 36 | 6 | 4 | 2 | 4 | 4 | 2 | 8 | 438 | 580 |
| 37 | 6 | 4 | 3 | 3 | 1 | 6 | 4 | 457 | 476 |
| 38 | 6 | 4 | 3 | 3 | 2 | 5 | 5 | 452 | 504 |
| 39 | 6 | **4** | 3 | **3** | **3** | **4** | 6 | 447 | 532 |
| 40 | 6 | 4 | 3 | 3 | 4 | 3 | 7 | 442 | 560 |
| 41 | 6 | 4 | 3 | 3 | 5 | 2 | 8 | 437 | 588 |
| 42 | 6 | 4 | 4 | 2 | 2 | 6 | 4 | 456 | 484 |
| 43 | 6 | 4 | 4 | 2 | 3 | 5 | 5 | 451 | 512 |
| 44 | 6 | **4** | 4 | **2** | **4** | **4** | 6 | 446 | 540 |
| 45 | 6 | 4 | 4 | 2 | 5 | 3 | 7 | 441 | 568 |
| 46 | 6 | 4 | 4 | 2 | 6 | 2 | 8 | 436 | 596 |
| **47** | **7** | **3** | **0** | **7** | **0** | **3** | **7** | **445** | **536** |
| **48** | **7** | **3** | **0** | **7** | **1** | **2** | **8** | **440** | **564** |
| 49 | 7 | 3 | 1 | 6 | 0 | 4 | 6 | 449 | 516 |
| 50 | 7 | 3 | 1 | 6 | 1 | 3 | 7 | 444 | 544 |
| 51 | 7 | 3 | 1 | 6 | 2 | 2 | 8 | 439 | 572 |
| 52 | 7 | 3 | 2 | 5 | 0 | 5 | 5 | 453 | 496 |
| 53 | 7 | 3 | 2 | 5 | 1 | 4 | 6 | 448 | 524 |
| 54 | 7 | 3 | 2 | 5 | 2 | 3 | 7 | 443 | 552 |

*continua na próxima página*

*continuação da página anterior*

| Nº | $x_{12}$ | $x_{13}$ | $x_{23}$ | $x_{24}$ | $x_{34}$ | $x_{35}$ | $x_{45}$ | $y_1$ | $y_2$ |
|---|---|---|---|---|---|---|---|---|---|
| 55 | 7 | 3 | 2 | 5 | 3 | 2 | 8 | 438 | 580 |
| 56 | 7 | 3 | 3 | 4 | 0 | 6 | 4 | 457 | 476 |
| 57 | 7 | 3 | 3 | 4 | 1 | 5 | 5 | 452 | 504 |
| 58 | 7 | 3 | 3 | 4 | 2 | 4 | 6 | 447 | 532 |
| 59 | 7 | 3 | 3 | 4 | 3 | 3 | 7 | 442 | 560 |
| 60 | **7** | **3** | 3 | 4 | **4** | **2** | 8 | 437 | 588 |
| 61 | 7 | 3 | 4 | 3 | 1 | 6 | 4 | 456 | 484 |
| 62 | 7 | 3 | 4 | 3 | 2 | 5 | 5 | 451 | 512 |
| 63 | 7 | 3 | 4 | 3 | 3 | 4 | 6 | 446 | 540 |
| 64 | 7 | 3 | 4 | 3 | 4 | 3 | 7 | 441 | 568 |
| 65 | 7 | 3 | 4 | 3 | 5 | 2 | 8 | 436 | 596 |
| 66 | 8 | 2 | 1 | 7 | 0 | 3 | 7 | 444 | 544 |
| 67 | 8 | 2 | 1 | 7 | 1 | 2 | 8 | 439 | 572 |
| 68 | 8 | 2 | 2 | 6 | 0 | 4 | 6 | 448 | 524 |
| 69 | 8 | 2 | 2 | 6 | 1 | 3 | 7 | 443 | 552 |
| 70 | 8 | 2 | 2 | 6 | 2 | 2 | 8 | 438 | 580 |
| 71 | 8 | 2 | 3 | 5 | 0 | 5 | 5 | 452 | 504 |
| 72 | **8** | **2** | 3 | **5** | **1** | **4** | 6 | 447 | 532 |
| 73 | 8 | 2 | 3 | 5 | 2 | 3 | 7 | 442 | 560 |
| 74 | 8 | 2 | 3 | 5 | 3 | 2 | 8 | 437 | 588 |
| **75** | **8** | **2** | **4** | **4** | **0** | **6** | **4** | **456** | **484** |
| 76 | 8 | 2 | 4 | 4 | 1 | 5 | 5 | 451 | 512 |
| 77 | 8 | 2 | 4 | 4 | 2 | 4 | 6 | 446 | 540 |
| 78 | 8 | 2 | 4 | 4 | 3 | 3 | 7 | 441 | 568 |
| 79 | **8** | **2** | 4 | **4** | **4** | **2** | 8 | 436 | 596 |
| 80 | 9 | 1 | 2 | 7 | 0 | 3 | 7 | 443 | 552 |
| 81 | 9 | 1 | 2 | 7 | 1 | 2 | 8 | 438 | 580 |
| 82 | 9 | 1 | 3 | 6 | 0 | 4 | 6 | 447 | 532 |
| 83 | 9 | 1 | 3 | 6 | 1 | 3 | 7 | 442 | 560 |
| 84 | 9 | 1 | 3 | 6 | 2 | 2 | 8 | 437 | 588 |
| 85 | 9 | 1 | 4 | 5 | 0 | 5 | 5 | 451 | 512 |
| 86 | 9 | 1 | 4 | 5 | 1 | 4 | 6 | 446 | 540 |
| 87 | 9 | 1 | 4 | 5 | 2 | 3 | 7 | 441 | 568 |
| 88 | 9 | 1 | 4 | 5 | 3 | 2 | 8 | 436 | 596 |
| **89** | **10** | **0** | **3** | **7** | **0** | **3** | **7** | **442** | **560** |
| **90** | **10** | **0** | **3** | **7** | **1** | **2** | **8** | **437** | **588** |
| **91** | **10** | **0** | **4** | **6** | **0** | **4** | **6** | **446** | **540** |
| 92 | 10 | 0 | 4 | 6 | 1 | 3 | 7 | 441 | 568 |
| **93** | **10** | **0** | **4** | **6** | **2** | **2** | **8** | **436** | **596** |

Temos ainda pontos extremos como imagens de pontos não extremos associados a soluções não-básicas como, por exemplo, o ponto extremo mais à esquerda que é imagem das soluções não-básicas 46, 65, 79 e 88.

Relativamente às restantes soluções não-dominadas, temos pontos não-extremos que são imagens de pontos extremos. Por exemplo, as soluções 47 e 90 são soluções

básicas admissíveis extremas, representadas por pontos extremos em $conv(X^I)$ e as suas imagens são soluções não-dominadas suportadas, mas não-extremas. Podemos, por isso, dizer que nem toda a solução eficiente extrema tem como imagem uma solução não-dominada extrema. Podemos ainda ter soluções eficientes extremas que têm como imagens soluções não-dominadas não-suportadas. No exemplo que temos estado a considerar a solução 89 é eficiente extrema, mas a sua imagem é uma solução não-dominada não-suportada.

Outra característica, que se torna bastante importante, uma vez que não foi considerada por muitos investigadores, como referido no Capítulo 3, é que nem todas as soluções não-dominadas suportadas são imagens de soluções eficientes extremas ou intermédias. Mostramos este aspecto nos dois exemplos seguintes. Consideremos, em primeiro lugar, o problema FRIBO da Figura 6.3. Uma parte da região admissível, no espaço dos objectivos, está representada na Figura 6.4.

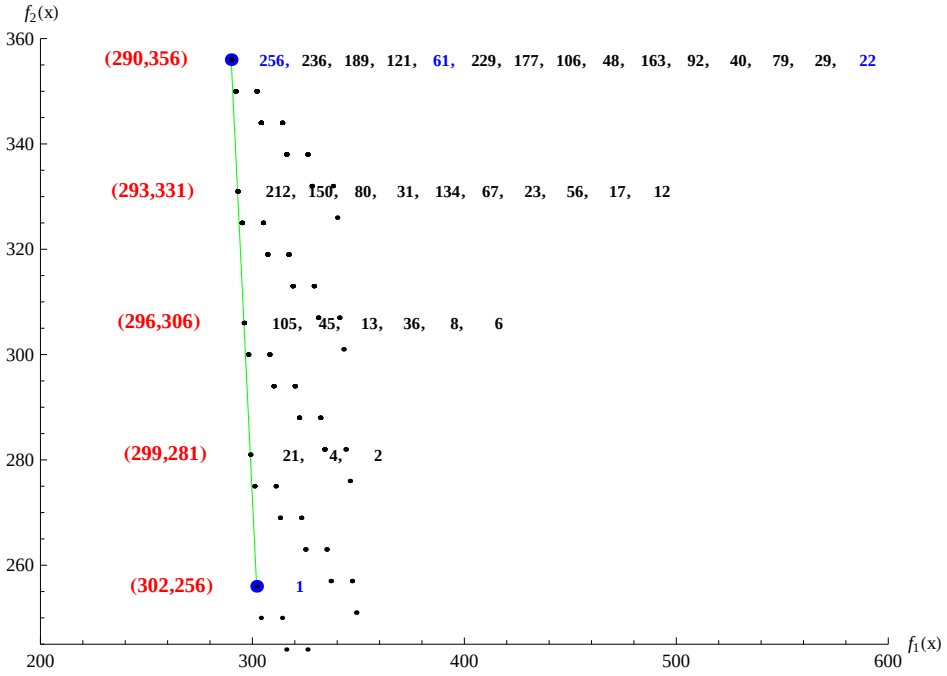



**Figura 6.4:** *Parte da região admissível no espaço dos objectivos.*

Repare-se, por exemplo, no ponto $(290, 356)$. Este ponto é imagem de 15 soluções, 3 das quais, as soluções 256, 61 e 22, são básicas admissíveis, isto é, são representadas por pontos extremos no poliedro $conv(X^I)$. Calculámos todas as soluções intermédias daquelas 3 soluções e verificámos que algumas das soluções que

têm como imagem este ponto não são intermédias. Assim, quando passamos da solução 256 para a 61 através de uma operação de *pivotação* são obtidas as soluções intermédias 236, 189 e 121; quando passamos da solução 256 para a 22 são obtidas as soluções intermédias 229, 163 e 79; e quando passamos da 61 para a 22 são obtidas as soluções 48, 40 e 29. Desta forma as soluções 236, 189, 121, 229, 163, 79, 48, 40 e 29 são soluções intermédias e as soluções 177, 106 e 92 são soluções não-básicas não-intermédias. Neste caso, todas as soluções têm a mesma imagem e havia, por isso, várias formas de a obter.

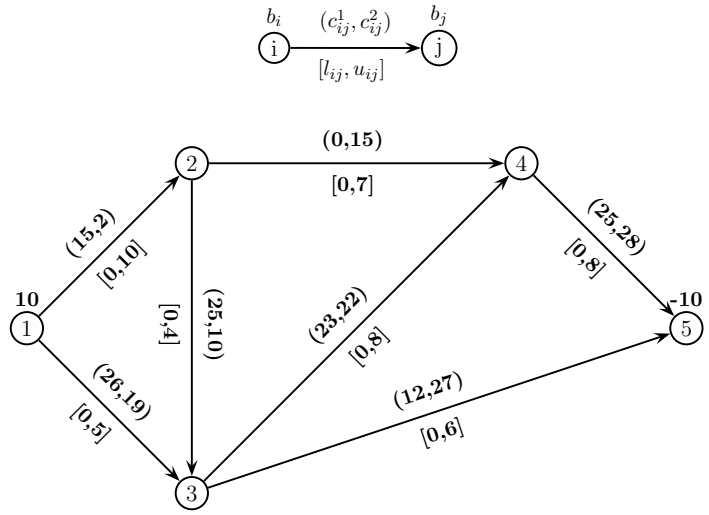



**Figura 6.5:** *Problema FRIBO.*

Vejamos agora um outro exemplo, onde a determinação das imagens das soluções extremas e intermédias deixa algumas das soluções não-dominadas suportadas por identificar. Consideremos o problema FRIBO representado na Figura 6.5. O conjunto das soluções não-dominadas suportadas está representado na Figura 6.6. As soluções admissíveis são as mesmas do primeiro exemplo, isto é, do problema FRIBO da Figura 6.1. Neste caso cada solução não-dominada é imagem apenas de uma solução do espaço das variáveis de decisão. As soluções 1, 47, 5, 89, 75 e 91 são básicas admissíveis sendo as soluções 89 e 91 degeneradas. A solução 1 tem como soluções adjacentes as soluções 47, 5 e 4, mas apenas as soluções 47 e 5 são suportadas. Quando passamos da solução 1 para a adjacente 47 é obtida a solução 25 como intermédia e quando passamos da solução 1 para a 5 não existem soluções intermédias. A solução 47 tem como soluções adjacentes as soluções 89, 1 e 48. A solução 48 não é eficiente. Quando passamos da solução 47 para a 89 temos como

soluções intermédias 66 e 80. A solução 5 tem como adjacentes as soluções 1, 75 e 20. A solução 20 não pertence ao conjunto das eficientes. Entre as soluções 5 e 75 existem as soluções intermédias 32 e 56. Podemos em conclusão dizer que entre os pontos representados 1 e 75 as soluções não-dominadas suportadas que são imagens de intermédias são as designadas por 25, 66, 80, 32 e 56. Desta forma as soluções 28 e 49 apesar de não-dominadas e suportadas não são imagens de soluções básicas admissíveis ou de soluções intermédias. Para visualizarmos melhor o tipo de soluções neste exemplo consideremos a representação das soluções eficientes. Este problema (Figura 6.5) pode ser formulado da seguinte forma:

$$
\begin{aligned}
&\text{“minimizar”} \quad (15x_{12} + 26x_{13} + 25x_{23} + 23x_{34} + 12x_{35} + 25x_{45},\\
&\qquad\qquad\quad\; 2x_{12} + 19x_{13} + 10x_{23} + 15x_{24} + 22x_{34} + 27x_{35} + 28x_{45}),\\
&\text{sujeito a:}\quad
\begin{array}{rrrrrrrl}
x_{12} & +x_{13} & & & & & & = 10,\\
-x_{12} & & +x_{23} & +x_{24} & & & & = 0,\\
& -x_{13} & -x_{23} & & +x_{34} & +x_{35} & & = 0,\\
& & & -x_{24} & -x_{34} & & +x_{45} & = 0,\\
& & & & & x_{35} & +x_{45} & = 10,
\end{array}\\
&\quad 0 \leqslant x_{12} \leqslant 10,\ 0 \leqslant x_{13} \leqslant 5,\ 0 \leqslant x_{23} \leqslant 4,\ 0 \leqslant x_{24} \leqslant 7,\\
&\quad 0 \leqslant x_{34} \leqslant 8,\ 0 \leqslant x_{35} \leqslant 6,\ 0 \leqslant x_{45} \leqslant 8 \text{ e}\\
&\quad x_{12},\ x_{13},\ x_{23},\ x_{24},\ x_{34},\ x_{35},\ x_{45} \text{ inteiros.}
\end{aligned}
$$

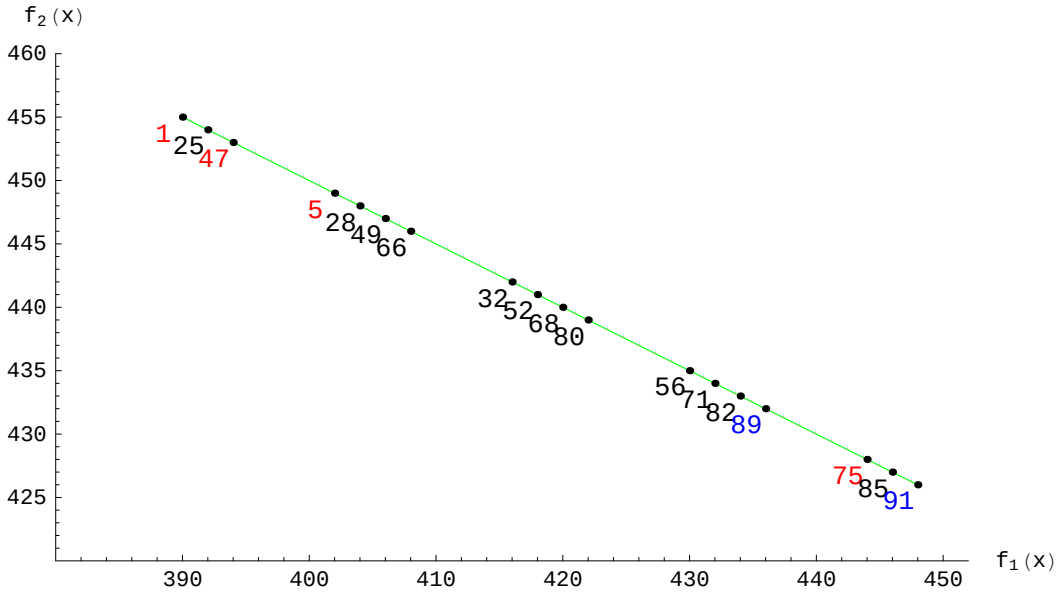



**Figura 6.6:** *Representação do conjunto das soluções não-dominadas suportadas.*

Consideremos a representação em $\mathbb{R}^3$ da região admissível deste problema, de acordo com a Subsecção 2.4. Para isso notemos que o problema dado é equivalente ao seguinte problema de programação linear inteira:

$$
\begin{array}{lll}
\text{“minimizar”} & (15x_{12} + 26x_{13} + 25x_{23} + 23x_{34} + 12x_{35} + 25x_{45}, & \\
& 2x_{12} + 19x_{13} + 10x_{23} + 15x_{24} + 22x_{34} + 27x_{35} + 28x_{45}), & \\
\text{sujeito a:} & x_{12} + x_{13} = 10, & \\
& x_{13} + x_{23} + x_{24} = 10, & \\
& x_{24} + x_{34} + x_{35} = 10, & \\
& x_{24} + x_{34} - x_{45} = 0, &
\end{array}
$$

$0 \leqslant x_{12} \leqslant 10$, $0 \leqslant x_{13} \leqslant 5$, $0 \leqslant x_{23} \leqslant 4$, $0 \leqslant x_{24} \leqslant 7$,
$0 \leqslant x_{34} \leqslant 8$, $0 \leqslant x_{35} \leqslant 6$, $0 \leqslant x_{45} \leqslant 8$ e
$x_{12}$, $x_{13}$, $x_{23}$, $x_{24}$, $x_{34}$, $x_{35}$, $x_{45}$ inteiros.

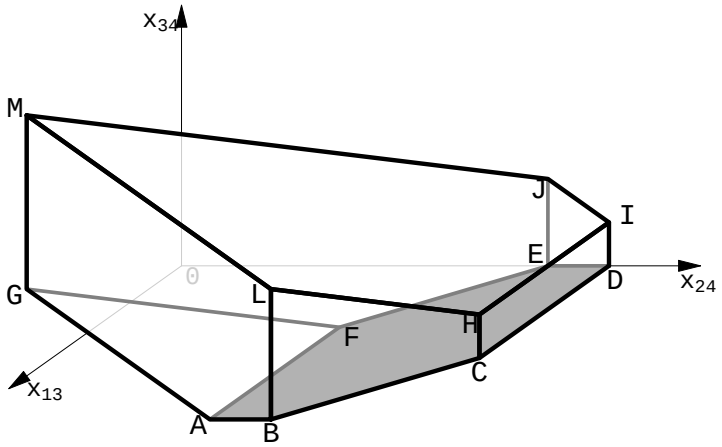



**Figura 6.7:** *Sólido representando a envolvente convexa da região admissível do problema na Figura 6.5.*

Consideremos as variáveis $x_{12}$, $x_{23}$, $x_{35}$ e $x_{45}$ como variáveis de folga na primeira, segunda, terceira e quarta restrições, respectivamente. A região admissível pode ser representada em $\mathbb{R}^3$ usando apenas as três variáveis de decisão $x_{13}$, $x_{24}$ e $x_{34}$. A envolvente convexa da região admissível é o poliedro representado na Figura 6.7. O conjunto das soluções eficientes suportadas está representado na Figura 6.8 pelos pontos numerados de acordo com a notação utilizada anteriormente. Nesta figura vêem-se claramente os pontos extremos e os pontos intermédios, nos segmentos de recta entre dois pontos extremos. Os pontos 28, 52, 49, 71, 68 e 82 são pontos que

não estão associados com soluções extremas nem com soluções intermédias, mas, no entanto, são soluções suportadas porque as suas imagens são pontos da fronteira de $Y^{\trianglerighteq}$.

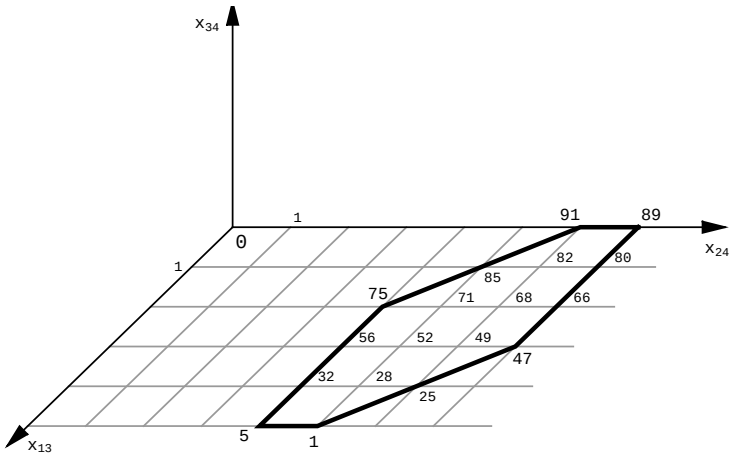



**Figura 6.8:** *Conjunto de soluções eficientes.*

## 6.3 Algoritmo de circuito de custo nulo

Nesta secção introduzimos os conceitos teóricos que servem de suporte ao novo algoritmo proposto. Consideremos o problema FICM associado à rede $\mathcal{N} = (\mathcal{G}, c, l, u, b)$. A optimalidade de uma solução, $x^*$, para este problema pode ser avaliada através do custo dos circuitos numa rede residual. Como foi visto na Secção 2.5, sabe-se que a rede residual $\mathcal{N}(x^*)$, associada à solução óptima $x^*$ do problema FICM, não contém qualquer circuito de custo negativo. Definimos custo reduzido (2.5) de um arco $(i, j)$ como a quantidade $\bar{c}_{ij} = c_{ij} - \pi_i + \pi_j$. Esta definição é aplicável tanto à rede residual como à rede associada ao problema inicial. A solução $x^*$ é uma solução óptima para o problema FICM se e só se $\bar{c}_{ij} \geqslant 0$, para todo o arco $(i, j)$ em $\mathcal{N}(x^*)$.

**Definição 6.3.1** *Designemos por $x'$ e $x''$ duas soluções admissíveis do problema FICM. A solução $x''$ diz-se uma solução de ciclo adjacente de $x'$ se $x''$ pode ser obtida a partir de $x'$ acrescentando $\delta$ unidades de fluxo num ciclo em $\mathcal{N}$, no sentido do arco $(p, q)$, associado a um circuito $W$ em $\mathcal{N}(x')$, onde $\delta = r_{pq} = \min\{r_{ij} : (i, j) \in W\}$. A solução obtida através do aumento de $\delta_1$ unidades de fluxo neste ciclo, onde $0 < \delta_1 < \delta$ com $\delta_1$ inteiro, designa-se por solução de ciclo intermédia de $(x', x'')$.*

**Proposição 6.3.1** *Se $x''$ é uma solução de ciclo adjacente de $x'$ então ou $x'$ é uma solução de ciclo adjacente de $x''$ ou $x'$ é uma solução de ciclo intermédia de $(x'', x''')$, onde $x'''$ é uma solução de ciclo adjacente de $x''$.*

*Demonstração:* Se $x''$ é uma solução de ciclo adjacente de $x'$ então existe um ciclo, $i_1 - a_1 - i_2 - a_2 \cdots - i_s - a_s - i_1$ ($a_k = (i_k, i_{k+1})$ ou $a_k = (i_{k+1}, i_k)$, $k = 1, 2, \cdots, s-1$ e $a_s = (i_s, i_1)$ ou $a_s = (i_1, i_s)$), na rede $\mathcal{N}$, tal que o aumento de $\delta$ unidades de fluxo ao longo deste ciclo leva à solução $x''$. Consideremos a solução $x''$ e o ciclo anterior com o sentido contrário, $i_1 - a_s - i_s \cdots - a_2 - i_2 - a_1 - i_1$. A mesma quantidade de fluxo enviada através deste ciclo origina a solução $x'$. Sendo $W$ o circuito correspondente na rede residual $\mathcal{N}(x'')$ e $\delta = \min\{r_{ij} : (i,j) \in W\}$ então $x'$ é uma solução de ciclo adjacente de $x''$. Se $\delta < \min\{r_{ij} : (i,j) \in W\}$, $x'$ é uma solução de ciclo intermédia de $(x'', x''')$, onde $x'''$ é uma solução de ciclo adjacente de $x''$.

□

**Proposição 6.3.2** *Consideremos dois pontos extremos adjacentes $x'$ e $x''$ do poliedro $X$. Se $x''$ é uma solução de ciclo adjacente de $x'$, então $x'$ é também uma solução de ciclo adjacente de $x''$.*

*Demonstração:* Sabemos que se $x'$ e $x''$ são soluções extremas adjacentes, então existe pelo menos um ciclo incremental em $\mathcal{N}$ que permite obter uma solução a partir da outra através de um aumento de fluxo ao longo deste ciclo. Estes ciclos estão associados a circuitos $W$ e $W'$ em $\mathcal{N}(x')$ e $\mathcal{N}(x'')$, respectivamente, e definem $x''$ como uma solução de ciclo adjacente de $x'$ e vice-versa.

□

Consideremos o problema paramétrico de fluxos em redes inteiro:

$$\begin{array}{ll} \text{minimizar} & \sum_{k=1}^{p} \lambda_k c^k x, \\ \text{sujeito a:} & x \in B^{(k)}, \end{array} \tag{6.1}$$

para algum $\lambda = (\lambda_1, \lambda_2, \cdots, \lambda_p) \in \Lambda$ e $B^{(k)} \subseteq X^I$.

No que se segue, assumiremos que o problema FRIMO tem mais que uma solução não-dominada suportada. Usamos a designação *sequência de ciclos* como uma sequência de soluções óptimas $x^{(1)}, x^{(2)}, \cdots, x^{(p)}$ para o problema (6.1), com $B^{(k)} = X^I$, tal que para cada par, $(x^{(q)}, x^{(q+1)})$, $x^{(q+1)}$ é uma solução de ciclo adjacente de $x^{(q)}$, $q = 1, 2, \cdots, p-1$, para algum problema do tipo (6.1). Uma solução

$x \in X^I$ diz-se estar numa sequência de ciclos $x^{(1)}, x^{(2)}, \cdots, x^{(p)}$ se $x$ é uma das soluções $x^{(1)}, x^{(2)}, \cdots, x^{(p)}$ ou se $x$ é uma solução de ciclo intermédia de $(x^{(q)}, x^{(q+1)})$, para algum $q = 1, 2, \cdots, p-1$.

As soluções não-dominadas suportadas estão na fronteira de $Y^{\geqq}$. Mostramos a seguir que o conjunto de todas as soluções não-dominadas suportadas para o problema FRIMO é conexo.

**Teorema 6.3.1** *Consideremos duas soluções não-dominadas suportadas e extremas $y' = f(x')$ e $y'' = f(x'')$ do problema FRIMO, na mesma face não-dominada máxima, $F_Y$, na fronteira de $Y^{\geqq}$. Qualquer solução não-dominada suportada em $F_Y$ é imagem de uma solução eficiente numa sequência de ciclos $x', \cdots, x''$.*

*Demonstração:* Para qualquer face eficiente máxima, $F_X$ de $X$, $f(F_X)$ é uma face não-dominada máxima de $Y$; e para cada face não-dominada máxima, $F_Y$ de $Y$, $f^{-1}(F_Y) \cap X$ é uma face eficiente máxima de $X$ (ver Dauer & Gallagher, 1996). Além disso, sabemos que os pontos de uma face eficiente máxima, $F_X$, são soluções óptimas do problema (6.1) para um valor fixo $\lambda^0 = (\lambda_1^0, \lambda_2^0, \cdots, \lambda_p^0) \in \Lambda$ (ver, por exemplo, Gal, 1977). Estamos apenas interessados nos pontos em $F_X$ e $F_Y$ com componentes inteiras.

A seguir mostramos que dada uma solução óptima do problema (6.1), com $B^{(k)} = X^I$, podemos determinar todas as soluções óptimas restantes em $X^I$ através dos circuitos de custo zero das redes residuais associadas às soluções já encontradas. Assumiremos, sem perda de generalidade, que entre dois nodos, $i$ e $j$, na rede $\mathcal{N}$, não existe mais do que um arco, o arco $(i, j)$ ou o arco $(j, i)$.

Dadas duas soluções óptimas $x^{(1)}$, $x^{(2)}$ de (6.1), com $\lambda = \lambda^0$ e $B^{(k)} = X^I$, ou $x^{(1)}, \cdots, x^{(2)}$ é uma sequência de ciclos ou $x^{(2)}$ está numa sequência de ciclos $x^{(1)}, \cdots, x^{(3)}$. De facto, a rede residual, $\mathcal{N}(x^{(1)})$, tem pelo menos um circuito de custo zero; caso contrário, considerando $x^{(1)}$ como o melhor fluxo (aquele que tem o menor custo) para este problema, o segundo melhor fluxo (calculado, por exemplo, usando o algoritmo da determinação do segundo melhor fluxo de Hamacher, 1995) teria um custo maior que o custo do fluxo $x^{(1)}$, mas isto significaria que o problema (6.1), com $B^{(k)} = X^I$, tinha apenas uma solução óptima, o que não é verdade, porque assumimos que os objectivos estão todos em conflito. Consideremos a solução de ciclo adjacente, $x^{(1,1)}$, de $x^{(1)}$ obtida através do circuito de custo zero $W^{(1)}$ em $\mathcal{N}(x^{(1)})$. Se $x^{(1,1)} = x^{(2)}$ ou $x^{(2)}$ é uma solução de ciclo intermédia de $(x^{(1)}, x^{(1,1)})$ então a demonstração termina. Caso contrário, consideremos o aumento de fluxo com $\delta = r_{pq} = \min\{r_{ij} : (i, j) \in W^{(1)}\}$ unidades que nos leva ao fluxo $x^{(1,1)}$. Consideremos a partição de $X^I$ em dois conjuntos, $B^{(1)}$ e $B^{(2)}$, de acordo com o sentido do ciclo correspondente na rede $\mathcal{N}$:

(1) se o seu sentido é o mesmo do arco $(p,q)$ na rede $\mathcal{N}$, consideremos o conjunto $B^{(1)} = \{x : x \in X^I \text{ e } l_{pq} \leqslant x_{pq} \leqslant a_{pq}\}$, onde $a_{pq}$ é o fluxo do arco $(p,q)$ na solução $x^{(1)}$;

(2) caso contrário, consideremos o conjunto $B^{(1)} = \{x : x \in X^I \text{ e } a_{qp} \leqslant x_{qp} \leqslant u_{qp}\}$, onde $a_{qp}$ é o fluxo do arco $(q,p)$ na solução $x^{(1)}$;

e o conjunto $B^{(2)} = X^I \setminus B^{(1)}$. Verifica-se que $x^{(1)} \in B^{(1)}$, $x^{(1,1)} \in B^{(2)}$ e $x^{(2)}$ está em $B^{(1)}$ ou $B^{(2)}$. Vejamos o que acontece em cada um destes dois casos:

a) Se $x^{(2)} \in B^{(1)}$, consideremos $x^{(1)}$ como a melhor solução do problema:

$$\begin{array}{ll} \text{minimizar} & \sum_{k=1}^{p} \lambda_k c^k x, \\ \text{sujeito a:} & x \in B^{(1)}, \end{array}$$

e a rede $\mathcal{N}$ associada com este problema. Determinemos a solução de ciclo adjacente, $x^{(1,2)}$, de $x^{(1)}$ através do circuito de custo zero $W^{(1,2)}$ em $\mathcal{N}(x^{(1)})$ ($x^{(1,2)}$ existe, uma vez que $x^{(2)}$ tem o mesmo custo que $x^{(1)}$). Se esta solução é $x^{(2)}$ ou se $x^{(2)}$ é uma solução de ciclo intermédia de $(x^{(1)}, x^{(1,2)})$ então a demonstração termina. Caso contrário, consideremos o aumento de fluxo de $r_{pq} = \min\{r_{ij} : (i,j) \in W^{(1,2)}\}$ unidades que conduz ao fluxo $x^{(1,2)}$. Consideremos a partição de $B^{(1)}$ nos subconjuntos $B^{(1,1)}$ e $B^{(1,2)}$ da mesma forma que a partição de $X^I$ foi feita nos conjuntos $B^{(1)}$ e $B^{(2)}$, em (1) e (2).

b) se $x^{(2)}$ está em $B^{(2)}$ consideremos $x^{(1,1)}$ como a melhor solução do problema:

$$\begin{array}{ll} \text{minimizar} & \sum_{k=1}^{p} \lambda_k c^k x, \\ \text{sujeito a:} & x \in B^{(2)}, \end{array}$$

e a nova rede $\mathcal{N}$ associada com este problema. Calculemos a solução de ciclo adjacente, $x^{(1,1,1)}$, de $x^{(1,1)}$ através do circuito de custo zero $W^{(1,1,1)}$ em $\mathcal{N}(x^{(1,1)})$. Se $x^{(1,1,1)} = x^{(2)}$ ou $x^{(2)}$ é uma solução de ciclo intermédia de $(x^{(1,1)}, x^{(1,1,1)})$ então a demonstração termina. Caso contrário, consideremos o aumento de fluxo de $r_{pq} = \min\{r_{ij} : (i,j) \in W^{(1,1,1)}\}$ unidades que conduz ao fluxo $x^{(1,1,1)}$. Consideremos a partição de $B^{(2)}$ nos subconjuntos $B^{(2,1)}$ e $B^{(2,2)}$ da mesma forma que foi feita a partição de $X^I$ em $B^{(1)}$ e $B^{(2)}$, em (1) e (2).

Repetindo este processo encontraremos uma sequência de ciclos $x^{(1)}, \cdots, x^{(3)}$ tal que $x^{(2)} = x^{(3)}$ ou tal que $x^{(2)}$ está na sequência de ciclos $x^{(1)}, \cdots, x^{(3)}$.

Concluímos que a solução óptima $x$ de (6.1), com $B^{(k)} = X^I$, está na sequência de ciclos $x', \cdots, x''$. De facto, sabemos que $x$ está numa sequência de ciclos $x', \cdots, x'''$

e a sequência de ciclos $x', \cdots, x''$ pode ser subdividida em duas sequências de ciclos $x', \cdots, x'''$ e $x''', \cdots, x''$. Assim a sequência de ciclos $x', \cdots, x''$ contém $x$ e a demonstração termina.

□

Podemos agora dizer que o conjunto de todas as soluções não-dominadas suportadas para o problema FRIMO é conexo, uma vez que o conjunto dos pontos extremos não-dominados também é conexo.

**Proposição 6.3.3** *O conjunto das soluções eficientes suportadas para um problema FRIMO é conexo.*

*Demonstração:* O conjunto das soluções eficientes extremas do problema FRMO é conexo como foi demonstrado, em 1983, por Warburton (1983). O Teorema 6.3.1 mostra que todas as soluções eficientes suportadas, numa mesma face, estão numa sequência de ciclos com origem numa solução eficiente extrema. Assim, concluímos que o conjunto de soluções eficientes suportadas para o problema FRIMO é conexo.

□

A demonstração do Teorema 6.3.1 dá origem ao Algoritmo 7 que determina todas as soluções eficientes/não-dominadas suportadas para o problema FRIMO. Consideremos o problema paramétrico (6.1), com $B^{(k)} = X^I$, para algum $\lambda$, fixo, associado à face eficiente máxima $F_X \in X$ e a solução não-dominada $y' = f(x')$ tal que $x' \in F_X$. O Teorema 6.3.1 diz que todas as soluções eficientes em $F_X$ para o problema FRIMO estão numa sequência de ciclos $x', ..., x''$.

Os valores dos $\lambda$'s para cada face eficiente são calculados através dos custos de um circuito na rede residual. Podemos começar por calcular a solução não-dominada associada com o mínimo do objectivo de ordem $p$ resolvendo um problema do tipo (6.1), com $B^{(k)} = X$, para valores de $\lambda_1, \lambda_2, \cdots, \lambda_{p-1}$ positivos e suficientemente próximos de 0 e tais que $\lambda_p = 1 - \lambda_1 - \lambda_2 - \cdots - \lambda_{p-1}$. Este problema pode ser resolvido utilizando o algoritmo de eliminação dos circuitos de custo negativo (Algoritmo 2). Designemos por $x^{(1)}$ a solução encontrada. Esta solução não tem circuitos de custo negativo e $\sum_{(i,j)\in W} \sum_{k=1}^{p} \lambda_k c_{ij}^k \geqslant 0$, para todo o circuito $W$ em $\mathcal{N}(x^{(1)})$. Este sistema de desigualdades permite a determinação de um vector $\lambda$ associado à face eficiente máxima contendo $x^{(1)}$. Gal (1977) mostra como isto pode ser feito para todas as faces eficientes máximas. Se tivermos apenas dois ou três objectivos, o procedimento seguinte pode ser usado.

*Algoritmo 7*: Algoritmo do circuito de custo zero.
{*Determina todas as soluções eficientes suportadas para o problema FRIMO.* }

---

**input:** Rede orientada $\mathcal{N} = (\mathcal{G}, (c^1, c^2, \cdots, c^p), u, b)$.
**output:** Conjunto de todas as soluções não-dominadas (eficientes) suportadas do problema FRIMO: $Y_s(X_s)$.

(1) **begin**
(2) Calcular o conjunto de todos os $\lambda$'s, $\lambda^{(1)}, \lambda^{(2)}, \cdots, \lambda^{(t)}$, associados às
(3) faces eficientes máximas $F_1, F_2, \cdots, F_t$ e um ponto extremo para cada
(4) face, $x^{(1)}, x^{(2)}, \cdots, x^{(t)}$, respectivamente;
(5) $E \leftarrow \{(x^{(1)}, \lambda^{(1)}, X^I), (x^{(2)}, \lambda^{(2)}, X^I), \cdots, (x^{(t)}, \lambda^{(t)}, X^I)\}$, onde $x^{(k)}$ é
(6) tal que $f(x^{(k)}) = y^{(k)}$, $k = 1, 2, \cdots, t$;
(7) $Y_s \leftarrow \{y^{(1')}, y^{(2')}, \cdots, y^{(t')}\}$;
(8) $X_s \leftarrow \{x^{(1'')}, x^{(2'')}, \cdots, x^{(t'')}\}$ (Estas soluções são as do conjunto $E$
(9) depois de eliminar as repetições);
(10) **while** $(E \neq \emptyset)$ **do**
(11) **begin**
(12) $E \leftarrow E \setminus \{(x^{(k)}, \lambda^{(k)}, B)\}$;
(13) Considerar o problema $\min_{x \in B} \sum_{k=1}^{p} \lambda_k c^k x$ para $\lambda = \lambda^{(k)}$ e a rede
(14) correspondente $\mathcal{N}$;
(15) **if** (existir uma solução de ciclo adjacente, $x^{(k')}$, de $x^{(k)}$ associada
(16) a um circuito de custo zero $W$ na rede residual $\mathcal{N}(x^{(k)})$ e
(17) com um arco $(l, q)$ tal que $r_{lq} = \min\{r_{ij} : (i, j) \in W\}$) **do**
(18) **begin**
(19) Juntar os elementos $(x^{(k)}, \lambda^{(k)}, B^{(1)})$ e $(x^{(k')}, \lambda^{(k)}, B^{(2)})$ ao con-
(20) junto $E$, onde $B^{(2)} = B \setminus B^{(1)}$ e $B^{(1)} = \{x \in B : 0 \leqslant x_{lq} \leqslant a_{lq}\}$
(21) se o arco $(l, q) \in \mathcal{N}$ ou $B^{(1)} = \{x \in B : a_{ql} \leqslant x_{ql} \leqslant u_{ql}\}$ caso
(22) contrário, onde $a_{lq}$ $(a_{ql})$ é o fluxo do arco $(l, q)$ $((q, l))$ na solu-
(23) ção $x^{(k)}$ ;
(24) **if** ($y^{(k')}$ não está em $Y_s$) **then** $Y_s \leftarrow Y_s \cup \{y^{(k')}\}$;
(25) **if** ($x^{(k')}$ não está em $X_s$) **then** $X_s \leftarrow X_s \cup \{x^{(k')}\}$;
(26) **for** ($x^{(k'')}$ solução de ciclo intermédia de $(x^{(k)}, x^{(k')})$) **do**
(27) **begin**
(28) **if** ($y^{(k'')}$ não pertence a $Y_s$) **then** $Y_s \leftarrow Y_s \cup \{y^{(k'')}\}$;
(29) **if** ($x^{(k'')}$ não pertence a $X_s$) **then** $X_s \leftarrow X_s \cup \{x^{(k'')}\}$;
(30) **end**
(31) **end**
(32) **end**
(33) **end**

---

Quando o problema tem dois objectivos, a Linha (2) do Algoritmo 7 pode começar pelo cálculo de todas as soluções não-dominadas suportadas extremas, isto é, os pontos extremos do conjunto $Y^{\geqq}$. Isto pode ser feito usando, por exemplo, uma pesquisa dicotómica ou um problema paramétrico resolvido através do algoritmo de ciclo-negativo ou através do algoritmo primal-dual (ver, por exemplo, Ehrgott, 2005 e Eusébio *et al.*, 2008). As faces não-dominadas máximas são, neste caso, segmentos de recta ligando pares de soluções não-dominadas adjacentes, uma vez que $Y$ apenas terá uma face não-dominada de dimensão 0 se o número de soluções não-dominadas extremas for igual a 1. Suponhamos que o declive de um segmento é $d$, então o vector $\lambda$ associado a esta face não-dominada máxima é $\lambda = (\frac{d}{d-1}, 1 - \frac{d}{d-1})$. Se o problema FRIMO tiver três objectivos então o problema pode ter faces não-dominadas máximas de dimensão 2 ou segmentos de recta. As faces de dimensão 2 são parte do plano cuja equação é $Ay_1 + By_2 + Cy_3 = D,\ A, B, C > 0$. Neste caso temos $\lambda = (\frac{A}{A+B+C}, \frac{B}{A+B+C}, \frac{C}{A+B+C})$. Se a face não-dominada máxima é um segmento de recta existem, em geral, vários vectores $\lambda$. Os candidatos são encontrados entre os vectores perpendiculares a este segmento de recta, de tal forma que $\lambda_k > 0, k = 1, 2, \cdots, p$ e $\sum_{k=1}^{p} \lambda_k = 1$.

## 6.4 Exemplo ilustrativo

Consideremos o problema FRIMO, com três objectivos, representado na Figura 6.9.

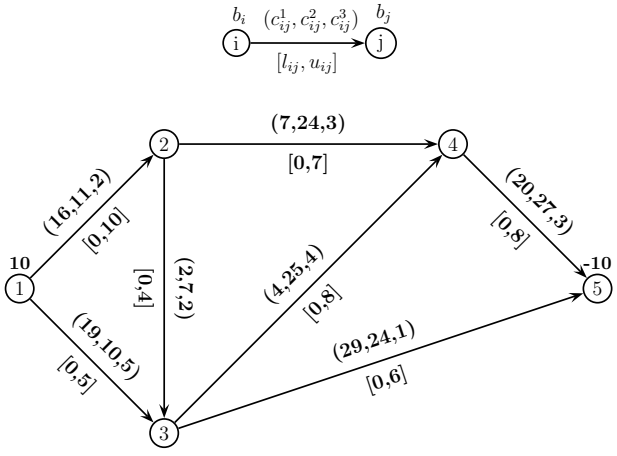



**Figura 6.9:** *Problema FRIMO.*

No primeiro passo do algoritmo, as faces eficientes máximas, para este problema,

são determinadas. Estas faces são os polígonos $[ABCDEF]$ e $[CDIH]$ e o segmento de recta $[IJ]$. Estas faces estão representadas na Figura 6.10 e 6.11 no espaço das variáveis de decisão e no espaço dos objectivos, respectivamente (ver os polígonos a cinzento). O ponto $C$ associado à solução eficiente $x^{(1)} = (7, 3, 0, 7, 0, 3, 7)$ pertence às duas primeiras faces e o ponto $I$ associado à solução $x^{(3)} = (10, 0, 3, 7, 1, 2, 8)$ pertence ao segmento de recta $[IJ]$. Os $\lambda$'s associados com estas faces são $\lambda^{(1)} = (\frac{44}{63}, \frac{1}{9}, \frac{4}{21})$, $\lambda^{(2)} = (\frac{76}{99}, \frac{1}{9}, \frac{4}{33})$ e $\lambda^{(3)} = (\frac{11}{13}, \frac{1}{13}, \frac{1}{13})$, respectivamente. Assim, temos, inicialmente, $E = \{(x^{(1)}, \lambda^{(1)}, X^I), (x^{(1)}, \lambda^{(2)}, X^I), (x^3, \lambda^{(3)}, X^I)\}$, $Y_s = \{y^{(1)}, y^{(3)}\} = \{(445, 536, 74), (442, 560, 71)\}$ e $X_s = \{x^{(1)}, x^{(3)}\} = \{(7, 3, 0, 7, 0, 3, 7), (10, 0, 3, 7, 1, 2, 8)\}$.

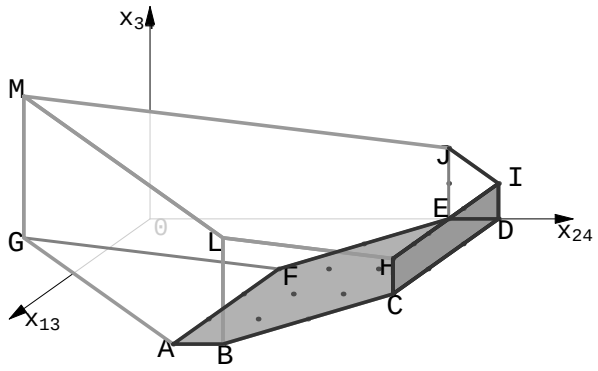



**Figura 6.10:** *Conjunto de soluções eficientes para o problema FRIMO na Figura 6.9.*

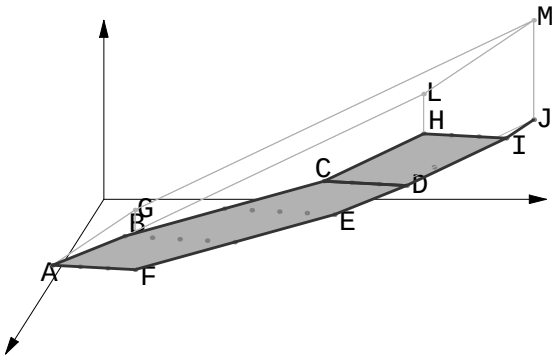



**Figura 6.11:** *Conjunto de soluções não-dominadas para o problema FRIMO na Figura 6.9.*

*It.*1: No segundo passo do algoritmo consideramos o elemento $(x^{(1)}, \lambda^{(1)}, X^I) = ((7,3,0,7,0,3,7), (\frac{44}{63}, \frac{1}{9}, \frac{4}{21}), X^I)$ em $E$ e colocamos

$$E \leftarrow \{(x^{(1)}, \lambda^{(2)}, X^I), (x^{(3)}, \lambda^{(3)}, X^I)\}.$$

Consideremos o problema auxiliar associado à primeira face.

$$\begin{array}{ll}
\text{minimizar} & \frac{44}{63}f_1(x) + \frac{1}{9}f_2(x) + \frac{4}{21}f_3(x) \\
& = \frac{115}{9}x_{12} + \frac{46}{3}x_{13} + \frac{23}{9}x_{23} + \frac{512}{63}x_{24} + \frac{19}{3}x_{34} + \frac{208}{9}x_{35} + \frac{1105}{63}x_{45}, \\
\text{sujeito a:} & \begin{array}{rrrrrrr}
x_{12} & +x_{13} & & & & & = 10, \\
-x_{12} & & +x_{23} & +x_{24} & & & = 0, \\
& -x_{13} & -x_{23} & & +x_{34} & +x_{35} & = 0, \\
& & & -x_{24} & -x_{34} & +x_{45} & = 0, \\
& & & & -x_{35} & -x_{45} & = -10,
\end{array} \\
& x_{12} \leqslant 10, x_{13} \leqslant 5, x_{23} \leqslant 4, x_{24} \leqslant 7, x_{34} \leqslant 8, x_{35} \leqslant 6, x_{45} \leqslant 8, \\
& x_{12}, x_{13}, x_{23}, x_{24}, x_{34}, x_{35}, x_{45} \geqslant 0 \text{ e inteiro.}
\end{array}$$

A rede residual associada à solução $x^{(1)}$ (Figura 6.12) está na Figura 6.13. Consideremos a solução de ciclo adjacente $x^{(4)} = (10,0,3,7,0,3,7)$ obtida através do ciclo de custo zero $1-2-3-1$. $(x^{(1)}, x^{(4)})$ tem $x^{(5)} = (8,2,1,7,0,3,7)$ e $x^{(6)} = (9,1,2,7,0,3,7)$ como soluções intermédias. As soluções não-dominadas correspondentes são $y^{(4)} = (442, 560, 71)$, $y^{(5)} = (444, 544, 73)$ e $y^{(6)} = (443, 552, 72)$.

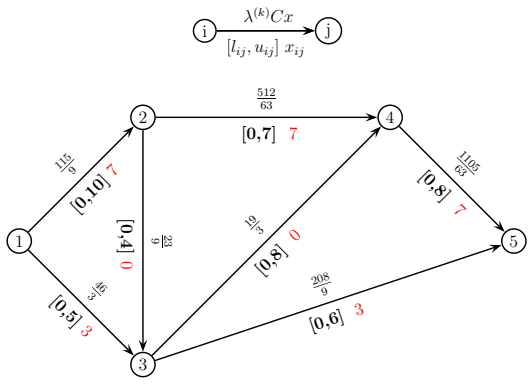



**Figura 6.12:** *Solução* $x^{(1)}$.

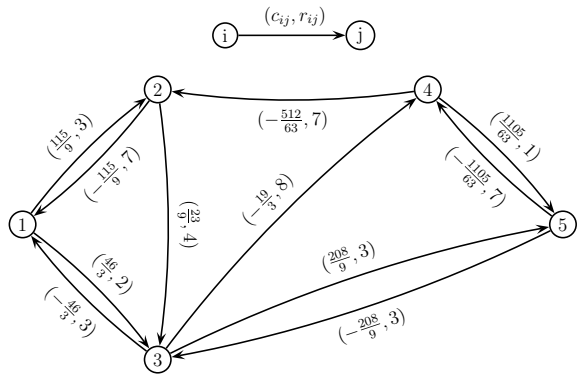



**Figura 6.13:** $\mathcal{N}(x^{(1)})$.

O novo conjunto $E$ é o seguinte:

$$\begin{aligned} E = \Big\{ & (x^{(1)}, \lambda^{(1)}, B^{(1)} = \{x \in X^I : 0 \leqslant x_{12} \leqslant 7\}), \\ & (x^{(4)}, \lambda^{(1)}, B^{(2)} = \{x \in X^I : 8 \leqslant x_{12} \leqslant 10\}), \\ & (x^{(1)}, \lambda^{(2)}, X^I), \\ & (x^{(3)}, \lambda^{(3)}, X^I) \Big\}. \end{aligned}$$

Temos, $Y_s = \{y^{(1)}, y^{(3)}, y^{(4)}, y^{(5)}, y^{(6)}\}$ e $X_s = \{x^{(1)}, x^{(3)}, x^{(4)}, x^{(5)}, x^{(6)}\}$

*It.*2: Como $E \neq \{\}$ o algoritmo continua considerando $(x^{(1)}, \lambda^{(1)}, B^{(1)}) \in E$. $E \leftarrow E \setminus \{(x^{(1)}, \lambda^{(1)}, B^{(1)})\}$.

A rede residual associada à solução $x^{(1)}$ (Figura 6.14), na rede associada com $B^{(1)}$, está representada na Figura 6.15. Consideremos a solução de ciclo adjacente $x^{(7)} = (5, 5, 0, 5, 0, 5, 5)$ obtida através do ciclo de custo zero $1-3-5-4-2-1$. $(x^{(1)}, x^{(7)})$ tem $x^{(8)} = (6, 4, 0, 6, 0, 4, 6)$ como solução intermédia. As soluções não-dominadas correspondentes são $y^{(7)} = (455, 480, 70)$ e $y^{(8)} = (450, 508, 72)$.

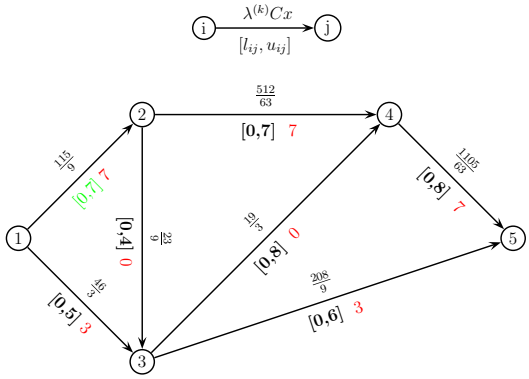



**Figura 6.14:** *Solução* $x^{(1)}$.

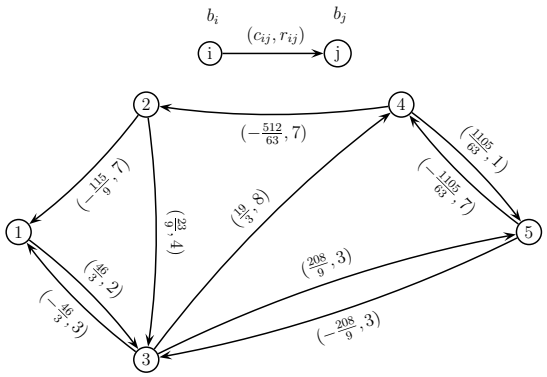



**Figura 6.15:** $\mathcal{N}(x^{(1)})$.

O novo conjunto $E$ é o seguinte:

$$
\begin{aligned}
E = \Big\{ & (x^{(1)}, \lambda^{(1)}, B^{(1,1)} = \{x \in X^I : 0 \leqslant x_{12} \leqslant 7 \text{ e } 0 \leqslant x_{13} \leqslant 3\}), \\
& (x^{(7)}, \lambda^{(1)}, B^{(1,2)} = \{x \in X^I : 0 \leqslant x_{12} \leqslant 7 \text{ e } 4 \leqslant x_{13} \leqslant 5\}), \\
& (x^{(4)}, \lambda^{(1)}, B^{(2)} = \{x \in X^I : 8 \leqslant x_{12} \leqslant 10\}), \\
& (x^{(1)}, \lambda^{(2)}, X^I), \\
& (x^{(3)}, \lambda^{(3)}, X^I) \Big\}.
\end{aligned}
$$

Os novos conjuntos $Y_s$ e $X_s$ são $Y_s = \{y^{(1)}, y^{(3)}, y^{(4)}, y^{(5)}, y^{(6)}, y^{(7)}, y^{(8)}\}$ e $X_s = \{x^{(1)}, x^{(3)}, x^{(4)}, x^{(5)}, x^{(6)}, x^{(7)}, x^{(8)}\}$.

*It.*3: Como $E \neq \{\}$ o algoritmo continua, considerando $(x^{(1)}, \lambda^{(1)}, B^{(1,1)}) \in E$. $E \leftarrow E \setminus \{(x^{(1)}, \lambda^{(1)}, B^{(1,1)})\}$.

Consideremos a solução de ciclo adjacente $x^{(9)} = (7, 3, 3, 4, 0, 6, 4)$ obtida através do ciclo de custo zero $2 - 3 - 5 - 4 - 2$ na rede associada com $B^{(1,1)}$. $(x^{(1)}, x^{(9)})$ tem $x^{(10)} = (7, 3, 1, 6, 0, 4, 6)$ e $x^{(11)} = (7, 3, 2, 5, 0, 5, 5)$ como soluções intermédias. As soluções não-dominadas correspondentes são $y^{(9)} = (457, 476, 65)$, $y^{(10)} = (449, 516, 71)$ e $y^{(11)} = (453, 496, 68)$.

O novo conjunto $E$ é

$$\begin{aligned} E = \Big\{ & (x^{(1)}, \lambda^{(1)}, B^{(1,1,1)} = \{x \in X^I : 0 \leqslant x_{12} \leqslant 7 \text{ e } 0 \leqslant x_{13} \leqslant 3 \text{ e } 0 \leqslant x_{35} \leqslant 3\}), \\ & (x^{(9)}, \lambda^{(1)}, B^{(1,1,2)} = \{x \in X^I : 0 \leqslant x_{12} \leqslant 7 \text{ e } 0 \leqslant x_{13} \leqslant 3 \text{ e } 4 \leqslant x_{35} \leqslant 6\}), \\ & (x^{(7)}, \lambda^{(1)}, B^{(1,2)} = \{x \in X^I : 0 \leqslant x_{12} \leqslant 7 \text{ e } 4 \leqslant x_{13} \leqslant 5\}), \\ & (x^{(4)}, \lambda^{(1)}, B^{(2)} = \{x \in X^I : 8 \leqslant x_{12} \leqslant 10\}), \\ & (x^{(1)}, \lambda^{(2)}, X^I), \\ & (x^{(3)}, \lambda^{(3)}, X^I) \Big\}. \end{aligned}$$

Os novos conjuntos $Y_s$ e $X_s$ são $Y_s = \{y^{(1)}, y^{(3)}, y^{(4)}, y^{(5)}, y^{(6)}, y^{(7)}, y^{(8)}, y^{(9)}, y^{(10)}, y^{(11)}\}$ e $X_s = \{x^{(1)}, x^{(3)}, x^{(4)}, x^{(5)}, x^{(6)}, x^{(7)}, x^{(8)}, x^{(9)}, x^{(10)}, x^{(11)}\}$.

*It.*4: Como $E \neq \{\}$ o algoritmo continua, considerando $(x^{(1)}, \lambda^{(1)}, B^{(1,1,1)}) \in E$ e $E = E \setminus \{(x^{(1)}, \lambda^{(1)}, B^{(1,1,1)})\}$.

Não existem circuitos de custo zero na rede residual associada com $B^{(1,1,1)}$.

*It.*5: Como $E \neq \{\}$ o algoritmo continua, considerando $(x^{(9)}, \lambda^{(1)}, B^{(1,1,2)}) \in E$. $E \leftarrow E \setminus \{(x^{(9)}, \lambda^{(1)}, B^{(1,1,2)})\}$.

Consideremos a solução de ciclo adjacente $x^{(10)} = (7, 3, 1, 6, 0, 4, 6)$ obtida através do ciclo de custo zero $2 - 4 - 5 - 3 - 2$ na rede associada com $B^{(1,1,2)}$. $(x^{(9)}, x^{(10)})$ tem $x^{(11)}$ como solução intermédia.

O novo conjunto $E$ é

$$\begin{aligned} E = \Big\{ & (x^{(9)}, \lambda^{(1)}, B^{(1,1,2,1)} = \{x \in X^I : 0 \leqslant x_{12} \leqslant 7 \text{ e } 0 \leqslant x_{13} \leqslant 3 \text{ e } x_{35} = 6\}), \\ & (x^{(10)}, \lambda^{(1)}, B^{(1,1,2,2)} = \{x \in X^I : 0 \leqslant x_{12} \leqslant 7 \text{ e } 0 \leqslant x_{13} \leqslant 3 \text{ e } 4 \leqslant x_{35} \leqslant 5\}), \\ & (x^{(7)}, \lambda^{(1)}, B^{(1,2)} = \{x \in X^I : 0 \leqslant x_{12} \leqslant 7 \text{ e } 4 \leqslant x_{13} \leqslant 5\}), \\ & (x^{(4)}, \lambda^{(1)}, B^{(2)} = \{x \in X^I : 8 \leqslant x_{12} \leqslant 10\}), \\ & (x^{(1)}, \lambda^{(2)}, X^I), \\ & (x^{(3)}, \lambda^{(3)}, X^I) \Big\}. \end{aligned}$$

Os conjuntos $Y_s$ e $X_s$ mantém-se os mesmos.

*It.*6: $(\cdots)$

As restantes soluções eficientes/não-dominadas são calculadas de forma similar.

## 6.5 Experiências computacionais, resultados e comentários

O algoritmo 7 foi implementado para problemas FRIBO, usando linguagem de programação C. O computador utilizado para os testes foi um *Intel Pentium* com um processador de 2.5GHz e 3GB RAM e sistema operativo *OS X*.

Com o objectivo de testar o algoritmo foram considerados cinco tipos de problemas resultantes da alteração do número de nodos, número de arcos, o total de oferta disponível e o número de nodos de oferta e de procura, de acordo com a Tabela 6.2. Para cada um destes tipos de problemas foram geradas, aleatoriamente, 30 instâncias utilizando a ferramenta *NETGEN*. Os coeficientes das funções objectivo foram aleatoriamente gerados (distribuição uniforme) entre o conjunto de números inteiros de 0 a 100. Todos os arcos têm capacidade limitada, isto é, os limites de fluxo em cada arco são finitos. Os parâmetros dos problemas estão de acordo com as convenções usadas no *NETGEN* (ver Klingman *et al.*, 1974).

**Tabela 6.2:** *Parâmetros dos problemas.*

| Tipo de problema | Nº de nodos | Nº de arcos | Nº origens | Nº destinos | Oferta total | % de arcos com $c_{ij}^1$ máximo | % de arcos com $c_{ij}^2$ máximo | Capacidade $u_{ij}$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 10 | 40 | 5 | 4 | 100 | 20 | 20 | 0 – 20 |
| 2 | 20 | 100 | 7 | 5 | 200 | 30 | 30 | 0 – 30 |
| 3 | 30 | 300 | 8 | 12 | 300 | 25 | 25 | 0 – 30 |
| 4 | 40 | 600 | 12 | 14 | 400 | 20 | 30 | 0 – 40 |
| 5 | 50 | 1000 | 15 | 15 | 600 | 25 | 25 | 0 – 40 |

Na Tabela 6.3, mostra-se o número médio de soluções não-dominadas suportadas (NDS), o número médio de soluções eficientes suportadas (ES) e o tempo médio *CPU*, em segundos, obtidos para cada problema.

**Tabela 6.3:** *Resultados obtidos.*

| Tipo de problema | Número de nodos | Número de arcos | Número médio de NDS | Número médio de ES | Tempo *CPU* médio (segundos) |
|---|---|---|---|---|---|
| 1 | 10 | 24 | 33,03 | 33,03 | 0,01 |
| 2 | 20 | 100 | 119,73 | 120,40 | 0,17 |
| 3 | 30 | 300 | 274,00 | 299,27 | 1,55 |
| 4 | 40 | 600 | 466,93 | 515,77 | 4,46 |
| 5 | 50 | 1000 | 759,83 | 951,90 | 14,11 |

O algoritmo parece ser capaz de determinar os conjuntos das soluções eficientes e não-dominadas suportadas para instâncias de tamanho pequeno ou médio num tempo *CPU* baixo. O tempo de cálculo de cada solução não-dominada suportada é em média de 0.007 segundos. Este tempo cresce com o número de nodos e com o número de arcos na rede, como pode ser visto na Tabela 6.4.

**Tabela 6.4:** *Tempo médio, em segundos.*

| Tipo de problema | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Tempo *CPU* (s) /NDS | 0.0004 | 0.0014 | 0.0057 | 0.0094 | 0.0186 |

# Capítulo 7

# Determinação de todas as soluções não-dominadas/eficientes

Neste capítulo começamos por fazer uma introdução, justificando a necessidade e descrevendo o aparecimento do algoritmo, para a determinação de todas as soluções eficientes/não-dominadas para o problema FRIBO. Revemos o método de restrição-$\varepsilon$ e as suas propriedades. Seguidamente, apresentamos o algoritmo. O algoritmo funciona sem quebrar a estrutura de rede para os problemas auxiliares, calculando uma solução óptima inteira do problema de restrição-$\varepsilon$. A exactidão deste algoritmo é também demonstrada. Apresentamos um exemplo de aplicação do algoritmo, ilustrado com as diversas regiões admissíveis do problema inicial e dos problemas auxiliares. Finalmente, apresentamos os resultados das experiências computacionais realizadas e fazemos uma análise estatística dos resultados, utilizando modelos de regressão linear.

## 7.1 Introdução

Neste capítulo descrevemos um algoritmo, designado de restrição-$\varepsilon$, que determina todas as soluções não-dominadas/eficientes, suportadas e não-suportadas, do problema FRIBO. O trabalho realizado e apresentado neste capítulo resulta da análise e melhoramento do algoritmo em Figueira (2000). Em Figueira (2000) o algoritmo está dividido em duas partes. Na primeira parte utiliza-se a estratégia proposta em Lee & Pulat (1993), para o cálculo de algumas soluções não-dominadas suportadas e, na segunda parte, são calculadas as restantes soluções não-dominadas. Verifica-se que o método proposto para calcular as soluções não-dominadas não-suportadas é capaz de calcular qualquer solução não-dominada e, deste modo, o algoritmo não precisa de utilizar duas fases evitando os erros cometidos em Lee & Pulat (1993). Assim, o algoritmo aqui proposto calcula todas as soluções não-dominadas ou eficientes sem qualquer divisão entre suportadas e não-suportadas. Procedemos também à implementação e experimentação deste algoritmo e os resultados obtidos são descritos neste capítulo.

## 7.2 Método da restrição-$\varepsilon$

O método da restrição-$\varepsilon$ é bastante conhecido na resolução de problemas de optimização. Este método foi introduzido por Haimes *et al.* (1971) e pode ser estudado mais pormenorizadamente em Chankong & Haimes (1983). Aqui, o método é aplicado para calcular soluções eficientes num problema FRIBO. Nesta secção apresentamos, resumidamente, algumas propriedades deste método.

Consideremos uma rede $\mathcal{N} = (\mathcal{G} = (\mathcal{V}, \mathcal{A}),\ (c^1,\ c^2),\ l,\ u,\ b)$ e o problema FRIBO associado, que pode ser escrito, matematicamente, como:

$$\begin{array}{lll} \text{minimizar} & f_1(x) = \displaystyle\sum_{(i,j)\in\mathcal{A}} c^1_{ij} x_{ij}, & \\ \text{minimizar} & f_2(x) = \displaystyle\sum_{(i,j)\in\mathcal{A}} c^2_{ij} x_{ij}, & \\ \text{sujeito a:} & \displaystyle\sum_{(k,j)\in\mathcal{A}} x_{kj} - \sum_{(i,k)\in\mathcal{A}} x_{ik} = b_k, & \forall k \in \mathcal{V}, \\ & 0 \leqslant x_{ij} \leqslant u_{ij},\ x_{ij} \text{ inteiro}, & \forall (i,j) \in \mathcal{A}. \end{array} \tag{7.1}$$

O problema de restrição-$\varepsilon$ associado ao problema (7.1) é o problema que consiste em minimizar um dos objectivos, sujeito ao conjunto de restrições para o problema original e mais uma restrição complementar relativamente ao outro objectivo, o qual não deverá ultrapassar um determinado valor real fixo. Matematicamente, o

problema de restrição-$\varepsilon$ pode ser formulado da seguinte forma:

$$\begin{array}{ll} \text{minimizar} & f_1(x), \\ \text{sujeito a:} & x \in X^I, \\ & f_2(x) \leqslant \varepsilon, \end{array} \tag{7.2}$$

onde $\varepsilon$ é um número real.

No problema de restrição-$\varepsilon$, $\varepsilon$ toma valores reais de forma a que este problema se mantenha admissível. Para identificar o conjunto de soluções eficientes/não-dominadas, é resolvida uma sequência de problemas do tipo (7.2), correspondentes a diferentes valores de $\varepsilon$ (ver Chankong & Haimes, 1983).

**Teorema 7.2.1** *Designemos por $y_1^*$ uma solução óptima para o problema (7.2), $X^* = \{x \in X^I : y_1^* = f_1(x)\}$ e $\bar{y}_2 = \min_{x \in X^*} f_2(x)$. Então $y' = (y_1^*, \bar{y}_2)$ é uma solução não-dominada para (7.1) (ver Chankong & Haimes, 1983).*

*Demonstração:* Suponhamos que $y'$ não é uma solução não-dominada para o problema (7.1). Então existe um outro $y = (y_1, y_2) \in Y^I$ tal que $y_1 \leqslant y_1^*$ e $y_2 < \bar{y}_2$ ou $y_1 < y_1^*$ e $y_2 \leqslant \bar{y}_2$. O último caso não pode ocorrer porque como $y_1 = f_1(x) < f_1(x')$ então $y_1^*$ não seria uma solução óptima para o problema (7.2), o que contraria, claramente, a nossa hipótese. O primeiro caso também não pode ocorrer. O valor de $y_1^*$ é óptimo para (7.2), então $f_1(x) \not< f_1(x')$. Assim, $f_1(x) = f_1(x')$, isto é, $x, x' \in X^*$ e $y_2 < \bar{y}_2$, o que resulta numa contradição, uma vez que $\bar{y}_2 = \min_{x \in X^*} f_2(x)$.

□

É importante realçar que este teorema é geral e não se exige, por exemplo, que o conjunto das soluções seja conexo. Este teorema será aplicado no algoritmo proposto, neste capítulo, para determinar todas as soluções eficientes e não-dominadas do problema (7.1).

## 7.3 Esboço do método

Nesta secção introduzimos um novo algoritmo para determinar o conjunto de todas as soluções não-dominadas de um problema FRIBO. O algoritmo recorre à resolução de uma sequência de problemas de restrição-$\varepsilon$ (7.2) e utiliza um procedimento PSAP, para a resolução deste problema não necessitando, por isso, de utilizar outro tipo de algoritmo para além dos algoritmos para a resolução de problemas de fluxos em redes. O algoritmo começa por calcular uma primeira solução não-dominada com o valor mínimo do primeiro objectivo e que, como é óbvio, é a solução não-dominada

com maior valor para o segundo objectivo. Depois determina as restantes soluções não-dominadas por ordem decrescente do valor do segundo objectivo e crescente do primeiro. O algoritmo pode também ser utilizado para determinar todas as soluções eficientes associadas com uma solução não-dominada. Bastará, para isso, explorar a árvore do PSAP até obter todas as soluções óptimas eficientes que são as óptimas alternativas para o problema (7.2). Neste caso, o algoritmo calcula todas as bases óptimas alternativas de forma a assegurar que as operações de *pivotação* conduzem à próxima base adjacente eficiente (ver Steuer, 1986, p. 122).

Este método depende em grande parte do cálculo das $EAG$s no algoritmo simplex para redes. Este algoritmo é suficientemente poderoso para evitar algumas das maiores dificuldades surgidas nos algoritmos baseados no simplex (ver Bazaraa *et al.*, 2005), principalmente devidas à degenerescência (devemos ter em conta, que se não existir degenerescência, o algoritmo simplex em redes converge num número finito de iterações; o termo degenerescência aqui significa que existe pelo menos um arco $(i, j)$ em $\mathcal{T}$, diferente do arco raiz, com fluxo igual a zero ou igual ao limite superior, $u_{ij}$.):

1. *Entrada em ciclo.* Sabemos que o algoritmo simplex para redes pode ser susceptível de entrar em ciclo, isto é, pode voltar infinitamente ao mesmo sítio através de uma sequência infinita de *pivots* degenerados que pode ocorrer. No entanto, existem regras que evitam a formação destes ciclos que são de fácil implementação e benéficas computacionalmente para problemas altamente degenerados, como é o caso dos problemas de fluxos em redes. Estas regras baseiam-se na manutenção de uma *estrutura de árvore geradora fortemente admissível* que consiste numa árvore geradora com um arco raiz na qual todos os arcos degenerados $(i, j)$ (se existir algum) estão orientados por forma a ser possível enviar uma quantidade positiva de fluxo de qualquer nodo para a raiz, através da árvore, sem violar qualquer limite do fluxo (ver Ahuja *et al.*, 1993). Nota: A impossibilidade da construção de tal árvore significaria que a rede actual poderia ser subdividida em duas ou mais subredes e o problema actual poderia ser subdividido em dois ou mais subproblemas associados a essas subredes.

2. *Número consecutivo de pivots.* Embora o uso de estruturas de árvore geradora fortemente admissíveis garantam convergência num número finito de iterações é, no entanto, possível que o número consecutivo de *pivots* degenerados se torne exponencial em $m$ e $n$. A escolha da variável de entrada através da regra da *Mais Recentemente Considerada* (MRC) garante que o número de *pivots* numa sequência de *pivots* degenerados, mantendo $\mathcal{T}$ com os mesmos arcos, não será superior a $n$ (ver Bazaraa *et al.*, 2005).

O algoritmo simplex em redes implementado é essencialmente baseado nos procedimentos propostos em Grigoriadis (1986), sendo tanto a entrada em ciclo como um número exponencial de *pivots* consecutivos evitados.

O algoritmo que propomos pode ser apresentado resumidamente da seguinte forma:

1. Calcular a solução correspondente ao mínimo lexicográfico $lex\ \min\{(f_1(x), f_2(x)) : x \in X\}$.
2. Calcular $\min\{f_2(x) : x \in X\}$.
3. Depois, o algoritmo calcula todas as soluções não-dominadas até atingir uma solução óptima do segundo objectivo, $f_2$. Estas soluções não-dominadas são obtidas resolvendo uma sequência de problemas de restrição-$\varepsilon$. Obtém-se, em primeiro lugar, a solução óptima do problema de restrição-$\varepsilon$ com variáveis reais e depois usa-se um PSAP. Se a solução óptima inteira do problema de restrição-$\varepsilon$ é única, então é eficiente para o problema bi-objectivo. Caso contrário, o algoritmo determina a solução óptima correspondente ao mínimo de $f_2(x)$ que é eficiente.

Apresentamos de seguida o procedimento geral. Depois mostramos como as soluções inteiras podem ser obtidas a partir das soluções não-inteiras. Finalmente, provamos a correcção do algoritmo.

### 7.3.1 Procedimento geral

Nesta secção, apresentamos o procedimento geral para determinar todas as soluções não-dominadas para o problema FRIBO. O algoritmo calcula todas as soluções não--dominadas começando com a solução não-dominada $y^{(0)} = (y_1^*, \bar{y}_2)$ associada ao mínimo lexicográfico ($y_1^* = \min_{x \in X} f_1(x)$ e $\bar{y}_2 = \min_{x^* \in X^*} f_2(x^*)$, onde $X^* = \{x^* : f_1(x^*) = y_1^*\}$) e termina com o mínimo do segundo objectivo $f_2(x)$. As restantes soluções não-dominadas são determinadas por ordem decrescente dos valores do segundo objectivo, usando problemas de restrição-$\varepsilon$. O algoritmo é apresentado a seguir como Algoritmo 8.

Na linha (3) do algoritmo determina-se a solução óptima lexicográfica $y^{(0)}$ para o problema FRIBO. Tal pode ser feito através do algoritmo simplex para redes, resolvendo o problema paramétrico em redes $\min_{x \in X} \lambda f_1(x) + (1-\lambda) f_2(x)$, sendo $\lambda$ um número real, no intervalo $]0, 1[$, suficientemente próximo de 1, de tal forma que a solução óptima encontrada seja também óptima para o problema $\min_{x \in X} f_1(x)$, isto é, os custos reduzidos $\lambda \bar{c}_{ij}^1 + (1-\lambda)\bar{c}_{ij}^2$ associados aos arcos $(i, j) \in L$ são não-negativos

e os associados aos arcos $(i,j) \in U$ são não-positivos para $\lambda \in [\lambda_1, \lambda_2]$ tal que $\lambda_1 < 1$ e $\lambda_2 \geqslant 1$. A existência de tal $\lambda$ está garantido pelas propriedades da programação linear (ver Zeleny, 1974), uma vez que estamos a supor que os mínimos lexicográficos do problema FRIBO existem e são diferentes.

*Algoritmo 8*: Algoritmo de restrição-$\varepsilon$.
{*Calcula todas as soluções não-dominadas do problema FRIBO.* }

---

**input:** Rede orientada $\mathcal{N} = (\mathcal{G} = (\mathcal{V}, \mathcal{A}), (c^1, c^2), u, b)$.
**output:** Conjunto de todas as soluções não-dominadas: $ND1$.

(1) **begin**
(2) $ND1 \leftarrow \{\}$;
(3) $y^{(0)} \leftarrow (y_1^*, \bar{y}_2) = lex \min_{x \in X} \begin{pmatrix} f_1(x) \\ f_2(x) \end{pmatrix}$;
(4) $y_2^* \leftarrow \min_{x \in X} f_2(x)$ ;
(5) $k \leftarrow 0$ e $ND1 \leftarrow ND1 \cup \{y^{(0)}\}$ ;
(6) $\varepsilon \leftarrow y_2^{(k)} - \rho$, sendo $\rho$ um número real positivo suficientemente pequeno,
(7) de tal forma que não exista qualquer solução não-dominada $y = (y_1, y_2)$
(8) tal que $y_2$ esteja entre $y_2^{(k)}$ e $y_2^{(k)} - \rho$ ;
(9)

$$y_1^{(k+1)} \leftarrow \begin{array}{rll} \text{minimizar} & f_1(x), & \\ \text{sujeito a:} & x \in X^I, & \text{(rest-}\varepsilon\text{)} \\ & f_2(x) \leqslant \varepsilon; & \end{array}$$

(10) $y_2^{(k+1)} \leftarrow \min_{x \in X^{(k+1)}} f_2(x)$, onde $X^{(k+1)} = \{x \in X^I : y_1^{(k+1)} = f_1(x)\}$ ;
(11) $ND1 \leftarrow ND1 \cup \{y^{k+1}\}$ ;
(12) **if** $(y_2^{(k+1)} = y_2^*)$ **then** STOP;
(13) **else** $k \leftarrow k + 1$;
(14) voltar à linha (6) ;
(15) **end**

---

Sabemos que, em geral, na linha (6) , $\rho \in ]0, 1]$, uma vez que não existe qualquer solução inteira $y = (y_1, y_2)$ tal que $y_2 \in ]y_2^{(k)} - 1, y_2^{(k)}[$. No entanto, poderá ser possível, para problemas FRIBO particulares, ter $\rho$ com um valor superior a 1, o que reduzirá a região de procura de soluções não-dominadas e aumentará a velocidade de cálculo.

Na secção seguinte, mostramos como pode ser feito o cálculo de uma solução inteira para o problema de restrição-$\varepsilon$, utilizando a estrutura de redes, não sendo necessário recorrer à programação linear, mais geral, para resolver este problema.

### 7.3.2 Cálculo de uma solução óptima inteira

As soluções óptimas para o problema (rest-$\varepsilon$) são obtidas através do PSAP com as componentes principais: separação e avaliação descritas no algoritmo abaixo.

Designemos por $Q$ uma lista de subproblemas do tipo $(K)$ a serem analisados e por *Apurada* a solução com o melhor valor para o subproblema (rest-$\varepsilon$) entre todos os que já foram analisados.

Passo 1. Considerar o problema inicial $(K)$ com $X^{(K)} = X^I$. Este problema representa a raiz da árvore do PSAP.

Fazer $Q = \{\}$ e $Apurada = (+\infty, +\infty)$

$$\begin{array}{ll} \text{“minimizar”} & (f_1(x), f_2(x)), \\ \text{sujeito a:} & x \in X^{(K)}. \end{array} \qquad (K)$$

Passo 2. Considerar os casos seguintes de acordo com o conjunto $X^{(K)}$.

a) Se existirem duas $EAG$s eficientes adjacentes suportadas, $EAG^{(k_1)}$ e $EAG^{(k_2)}$, tais que $y_2^{(k_1)} < \varepsilon \leqslant y_2^{(k_2)}$, fazer $x^{(k')} = x^{(k_1)} + \Delta_1 d$, onde $\Delta_1 = \Delta \times \frac{\varepsilon - y_2^{(k_1)}}{y_2^{(k_2)} - y_2^{(k_1)}}$, e $d = \frac{x^{(k_2)} - x^{(k_1)}}{\Delta}$. $\Delta$ é a quantidade de fluxo enviado ao longo do ciclo $\mathcal{C}$ numa iteração simplex, quando o arco $(k, l) \notin \mathcal{T}$ é inserido em $\mathcal{T}$, passando da $EAG^{(k_1)}$ para a $EAG^{(k_2)}$.

(i) Se todas as componentes da solução $x^{(k')}$ são inteiras ou $y_1^{(k')}$ é maior ou igual que a primeira componente de *Apurada*, actualizar *Apurada* (Avaliação).

(ii) Caso contrário, considerar os dois novos problemas $(K1)$ e $(K2)$ (Separação).

$$\begin{array}{ll} \text{“minimizar”} & (f_1(x), f_2(x)), \\ \text{sujeito a:} & x \in X^{(K)}, \\ & x_{kl} \leqslant \lfloor x_{kl}^{(k')} \rfloor \end{array} \qquad (K1)$$

e

$$\begin{array}{ll} \text{“minimizar”} & (f_1(x), f_2(x)), \\ \text{sujeito a:} & x \in X^{(K)}, \\ & x_{kl} \geqslant \lceil x_{kl}^{(k')} \rceil. \end{array} \qquad (K2)$$

Fazer $Q = Q \cup \{K1, K2\}$

b) Caso contrário, se existir uma $EAG$ eficiente suportada $EAG^{(k')}$ tal que $y_2^{(k')} < \varepsilon$ e não existir outra $EAG$ suportada eficiente $EAG''$ tal que $y_2'' > y_2^{(k')}$ então actualizar *Apurada*.

Passo 3. Se $Q = \{\}$, STOP; caso contrário, considerar outro problema $(K)$ de $Q$, fazer $Q = Q \setminus \{(K)\}$ e repetir o Passo 2.

No Passo 2. a actualização de *Apurada* é feita considerando $Apurada = y^{(k')}$ quando $y_1^{(k')} < Apurada_1$ ou se se verifica $y_1^{(k')} = Apurada_1$ e $y_2^{(k')} < Apurada_2$. A notação $\lfloor x_{kl}^{(k')} \rfloor$ representa o maior inteiro menor ou igual a $x_{kl}^{(k')}$ e $\lceil x_{kl}^{(k')} \rceil$ representa o menor inteiro maior ou igual a $x_{kl}^{(k')}$.

No Passo 2.a)(ii), quando o problema actual é dividido em dois subproblemas, a $EAG^{(k1)}$ é eficiente para um deles e a $EAG^{(k2)}$ para o outro. Isto permite-nos encontrar as duas soluções adjacentes no Passo 2.a) através de um caminho de soluções adjacentes eficientes. Sabemos que, dada uma $EAG$ eficiente, $EAG'$, para o problema FRIBO e um ponto eficiente extremo $\bar{x}$, começando em $EAG'$ e realizando apenas *pivotações* eficientes, isto é, passando de uma solução eficiente para outra solução adjacente eficiente, podemos obter uma $EAG$ associada com a solução $\bar{x}$ (ver Schechter & Steuer, 2005). Consideremos uma solução eficiente $x'$ e passemos para uma solução adjacente eficiente $x''$. As soluções $x'$ e $x''$ são óptimas para o problema paramétrico $\min\limits_{x \in X} \lambda f_1(x) + (1-\lambda) f_2(x)$ para $\lambda \in [\lambda_1, \lambda_2]$, onde $\lambda_1 = \min \Lambda_{12}$ e $\lambda_2 = \max \Lambda_{12}$,

$$\begin{aligned} \Lambda_{12} = \{ \quad & \lambda \in [0,1] : \lambda(\bar{c}_{ij}^1 - \bar{c}_{ij}^2) + \bar{c}_{ij}^2 \geqslant 0 \text{ se } (i,j) \in L \\ & \text{e } \lambda(\bar{c}_{ij}^1 - \bar{c}_{ij}^2) + \bar{c}_{ij}^2 \leqslant 0 \text{ se } (i,j) \in U \}, \end{aligned} \tag{7.3}$$

$\lambda_1$ é igual a 0 ou a $\frac{-\bar{c}_{k_1 l_1}^2}{\bar{c}_{k_1 l_1}^1 - \bar{c}_{k_1 l_1}^2}$ para algum arco $(k_1, l_1)$ que não está em $\mathcal{T}$ e $\lambda_2$ é igual a 1 ou $\frac{-\bar{c}_{k_2 l_2}^2}{\bar{c}_{k_2 l_2}^1 - \bar{c}_{k_2 l_2}^2}$ para algum arco $(k_2, l_2)$ que não está em $\mathcal{T}$. Se $\lambda_1 \in ]0,1[$, então existe $\lambda_1' \in [0,1]$, $\lambda_1' \leqslant \lambda_1$ tal que uma nova solução óptima é obtida para o problema paramétrico com $\lambda \in [\lambda_1', \lambda_1]$, quando o arco de entrada é $(k_1, l_1)$. Da mesma forma, se $\lambda_2 \in ]0,1[$, então existe um $\lambda_2' \in [0,1]$, $\lambda_2' \geqslant \lambda_2$, tal que uma nova solução óptima é obtida para este problema com $\lambda \in [\lambda_2, \lambda_2']$, quando o arco de entrada é $(k_2, l_2)$.

O procedimento termina com uma solução não-dominada $Apurada \neq (+\infty, +\infty)$, uma vez que para cada problema (rest-$\varepsilon$) existe pelo menos uma solução não-dominada tal que $f_2(x) \leqslant \varepsilon$.

### 7.3.3 Cálculo das soluções não-inteiras

Uma das principais características do PSAP proposto está relacionada com a variável de ramificação $x_{kl}$ que é a variável de entrada permitindo passar da $EAG^{(k1)}$ para a $EAG^{(k2)}$ (Passo 2). A solução óptima para o problema (rest-$\varepsilon$) é calculada no Passo 2 como $x^{(k')} = x^{(k_1)} + \Delta_1 d$, onde $d$ é um vector com a direcção da recta $x^{(k_1)}x^{(k_2)}$. Por construção, $x^{(k_1)}$ e $x^{(k_2)}$, são tais que $\varepsilon \in [f_2(x^{(k_1)}), f_2(x^{(k_2)})]$. Como $f_2$ é uma função linear sabemos que existe uma solução $x^{(k')}$ no segmento de recta $[x^{(k_1)}x^{(k_2)}]$ tal que $f_2(x^{(k')}) = \varepsilon$. Cada ponto de $[x^{(k_1)}x^{(k_2)}]$ pode ser escrito como $x^{(k_1)} + \Delta_1 d$, $\Delta_1 \in [0, \Delta]$, sendo $\Delta$ o valor no Passo 2. O valor de $\Delta_1$ correspondente a este ponto $x^{(k')}$ pode ser obtido considerando a semelhança entre os dois triângulos $[ACE]$ e $[BCD]$, na Figura 7.1, e a proporcionalidade entre os comprimentos dos seus lados: $\frac{\overline{EC}}{\overline{AC}} = \frac{\overline{DC}}{\overline{BC}}$. Note que o ponto $x^{(k')}$ poderia também ser $x^{(k_2)} + \Delta_1 d'$, onde $d' = \frac{x^{(k_1)} - x^{(k_2)}}{\Delta}$. A quantidade $\Delta_1$ é agora obtida considerando a semelhança entre os dois triângulos $[ACE]$ e $[ABF]$.

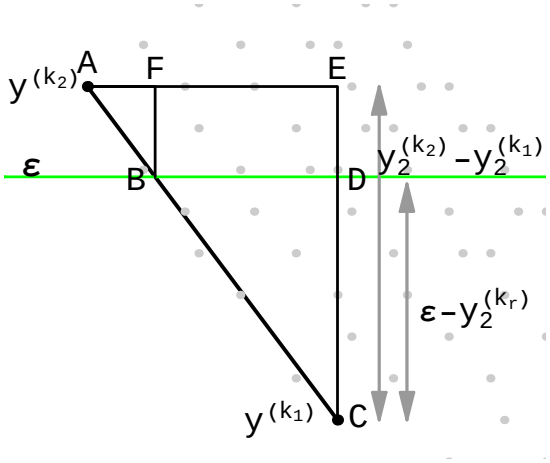



**Figura 7.1:** *Cálculo do valor* $\Delta_1$.

### 7.3.4 A exactidão do algoritmo

Para provar a exactidão do Algoritmo 8 temos de mostrar que todas as soluções não-dominadas serão encontradas e que nenhuma das soluções determinadas pelo algoritmo é uma solução dominada.

**Proposição 7.3.1** *No final da aplicação do Algoritmo 8, as soluções do conjunto* $ND1$ *são todas não-dominadas e qualquer solução admissível que não esteja em* $ND1$ *é dominada.*

*Demonstração:* Todas as soluções no conjunto $ND1$ são não-dominadas. De facto, a primeira solução encontrada é não-dominada uma vez que é um mínimo lexicográfico. As restantes soluções em $ND1$ são obtidas de acordo com o Teorema 7.2.1, sendo, por isso, soluções não-dominadas.

Suponhamos que existe uma solução não-dominada $y$ não pertencente a $ND1$. Então, por definição de solução não-dominada, não existe uma solução $y'$ tal que $y'_1 \leqslant y_1$ e $y'_2 < y_2$ ou $y'_1 < y_1$ e $y'_2 \leqslant y_2$. Representemos por $y^{(k+1)}$ a solução obtida de $y^{(k)}$ no Ponto (4) do Algoritmo 8 tal que $y_2^{(k+1)} \leqslant y_2 < y_2^{(k)}$. Como $y_1^{(k+1)} \leqslant y_1$ e $y_2^{(k+1)} \leqslant y_2$, então $y \geqq y^{(k+1)}$ e $y \neq y^{(k+1)}$, isto é, $y$ é uma solução dominada, o que nos conduz a uma contradição. O que significa que não pode existir qualquer solução não pertencente a $ND1$, como queríamos demonstrar.

□

## 7.4 Exemplo ilustrativo

Nesta secção ilustramos a forma como o algoritmo proposto funciona. Consideremos o problema FRIBO representado na Figura 7.2.

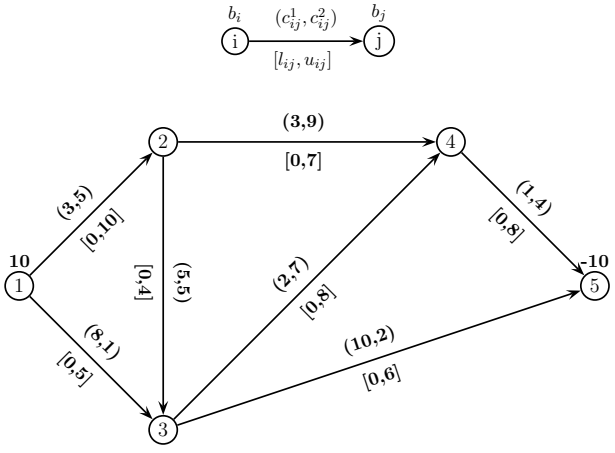



**Figura 7.2:** *Rede de um problema FRIBO.*

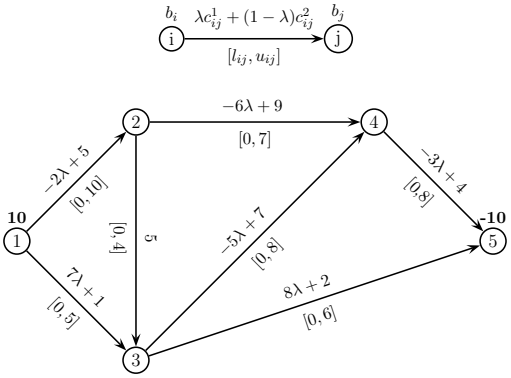



**Figura 7.3:** *Problema paramétrico.*

Este problema tem 93 soluções admissíveis (ver Tabela 6.1). Entre as 93 soluções, apenas 10 são eficientes. Estas 10 soluções correspondem biunivocamente a 10 soluções não-dominadas. Na Figura 7.11 estão representadas todas as soluções admissíveis, com as soluções não-dominadas representadas por um círculo.

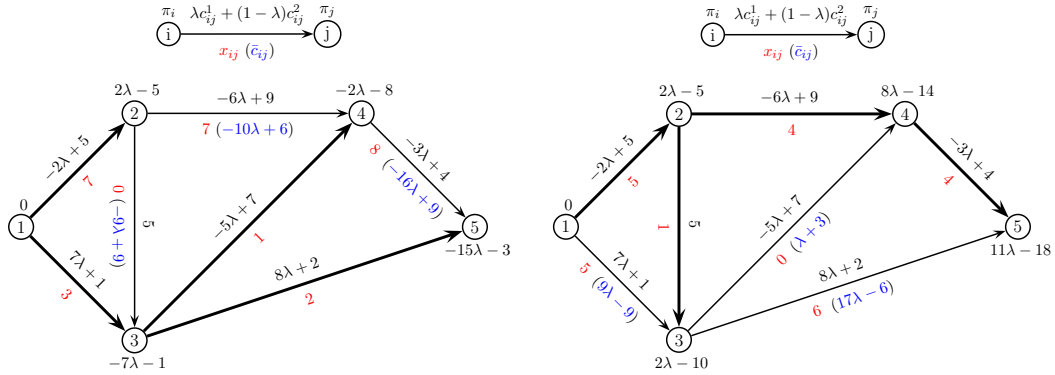



**Figura 7.4:** *Soluções (a) $x^{(0)} = x^{48}$; (b) $x^5$.*

Quando executamos o Algoritmo 8, o primeiro conjunto de soluções não-dominadas é $ND1 = \{\}$. De seguida, determinamos $y^{(0)}$ resolvendo o problema da Figura 7.3 com $\lambda \in ]0, 1[$ próximo de 1. A solução $x^{(0)} = (7, 3, 0, 7, 1, 2, 8)$ é obtida com $y^{(0)} = (f_1(x^{(0)}), f_2(x^{(0)})) = (96, 144) = y^{48}$. O valor óptimo para o problema $\min_{x \in X} f_2(x)$ é $y_2^* = 99$ (ver Figura 7.11). O conjunto actual de soluções não-dominadas passa a ser $ND1 = \{\} \cup \{y^{48}\} = \{y^{48}\}$.

O problema (rest-$\varepsilon$) é resolvido usando o procedimento descrito acima com o valor de $\varepsilon = 143.5$. A escolha de $\varepsilon$ poderia ter sido diferente. Sabemos que qualquer número real entre 143 e 144 serviria, uma vez que todas as componentes de qualquer solução admissível em $Y^I$, são inteiras. A solução não-dominada $y^{27} = (100, 138)$ é obtida. Actualizamos o conjunto das soluções não-dominadas obtidas para $ND1 = \{y^{27}, y^{48}\}$.

De seguida, precisamos resolver o problema (rest-$\varepsilon$). Consideremos $\varepsilon = 137.5$. A árvore do PSAP da Figura 7.10 contém todas as soluções visitadas antes de se atingir a solução óptima. Descrevemos de seguida as iterações utilizadas para determinar essa solução.

**<u>It</u> 1**

1. Resolver o problema (rest-$\varepsilon$) considerando primeiro o problema (K).

   "minimizar" $f(x) = (f_1(x), f_2(x))$,
   sujeito a: $x \in X^I$,

   onde $f_1(x) = 3x_{12}+8x_{13}+5x_{23}+3x_{24}+2x_{34}+10x_{35}+x_{45}$, $f_2(x) = 5x_{12}+x_{13}+5x_{23}+9x_{24}+7x_{34}+2x_{35}+4x_{45}$ e $X^I = \{(x_{12}, x_{13}, x_{23}, x_{24}, x_{34}, x_{35}, x_{45}) \in \mathbb{N}_0^7 : x_{12} + x_{13} = 10,\ -x_{12} + x_{23} + x_{24} = 0,\ -x_{13} - x_{23} + x_{34} + x_{35} = 0,\ -x_{24} - x_{34} + x_{45} = 0,\ -x_{35} - x_{45} = -10,\ x_{12} \leqslant 10,\ x_{13} \leqslant 5,\ x_{23} \leqslant 4,\ x_{24} \leqslant 7,\ x_{34} \leqslant 8,\ x_{35} \leqslant 6,\ x_{45} \leqslant 8\}$.

   Fazer $Q = \{\}$, $Apurada = (+\infty, +\infty)$.

2. $EAG^{48}$ e $EAG^4$ são $EAG$s adjacentes, eficientes e suportadas tais que $y_2^4 < 137.5 \leqslant y_2^{48}$. Descrevemos de seguida como se obtém $EAG^4$.

   A solução $x^{(0)} = x^{48}$ é óptima para o problema paramétrico com $\frac{3}{5} \leqslant \lambda \leqslant 1$. Consideremos a $EAG$, $EAG^{48}$, associada a esta solução. (ver Figura 7.4(a), os arcos em $\mathcal{T}$ estão com um traço mais espesso). Quando $\lambda = \frac{3}{5}$, uma nova solução alternativa é obtida inserindo o arco $(2, 4)$ na árvore $\mathcal{T}$ através de uma iteração do simplex (Figura 7.5). Isto leva-nos a uma nova solução $x^4 = (5, 5, 0, 5, 3, 2, 8)$ e a uma nova $EAG$, a $EAG^4$ com $y^4 = (f_1(x^4), f_2(x^4)) = (104, 132)$. As $EAG$s $EAG^{48}$ e $EAG^4$ são adjacentes e tais que $y_2^4 < 137.5 \leqslant y_2^{48}$.

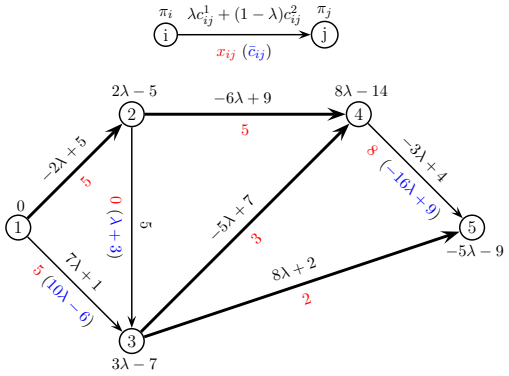



**Figura 7.5:** *Solução* $x^4$.

O arco $(1,3)$ é o arco de entrada que nos permite passar da $EAG^4$ para a sua adjacente $EAG^{48}$. O ponto $y^{(k')}$ está na intersecção das rectas $y_2 = \varepsilon$ e $y^4y^{48}$. Temos $y_1^{(k')} = f_1(x^{(k')}) = 100.33333$, onde $x^{(k')} = x^4 + \Delta_1 d = (5.91667,\ 4.08333,\ 0,\ 5.91667,\ 2.08333,\ 2,\ 8)$ com $\Delta_1 = 2 \times \frac{137.5-132}{144-132} = 0.916667$ e $d = (1, -1, 0, 1, -1, 0, 0)$.

A região admissível $X^{(K)}$ é dividida em dois subconjuntos: $X^{(k_1)} = X^{(K)} \cap \{x \in X : x_{13} \leqslant 4\}$ e $X^{(k_2)} = X^{(K)} \cap \{x \in X : x_{13} \geqslant 5\}$.

Consideremos os dois subproblemas:

$$\begin{array}{ll} \text{“minimizar”} & (f_1(x), f_2(x)), \\ \text{sujeito a:} & x \in X^{(K_1)}, \end{array} \qquad (B)$$

e

$$\begin{array}{ll} \text{“minimizar”} & (f_1(x), f_2(x)), \\ \text{sujeito a:} & x \in X^{(K_2)}. \end{array} \qquad (C)$$

Temos $Q = Q \cup \{B, C\} = \{B, C\}$. $EAG^{48}$ é uma $EAG$ eficiente suportada para o problema $B$ e $EAG^4$ é uma $EAG$ eficiente suportada para o problema $C$. Esta informação é guardada para ser utilizada mais tarde.

3. Consideramos seguidamente o problema (K)=$B$ (ver Figura 7.6). $Q = \{C\}$.

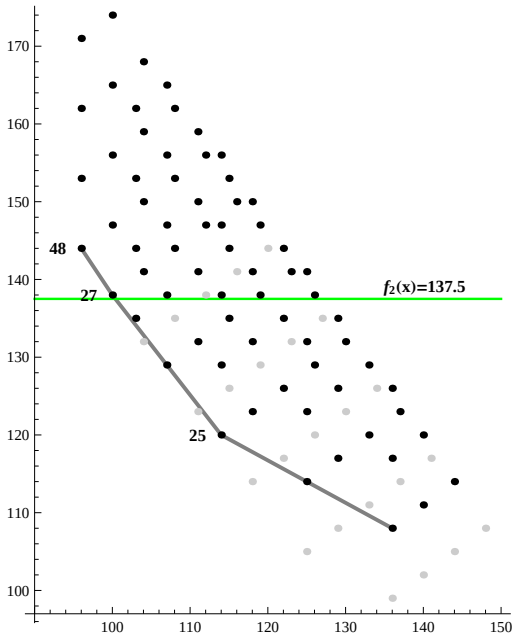



**Figura 7.6:** *Região admissível para o problema B.*

**It 2**

**2.** $EAG^{48}$ é uma $EAG$ eficiente e suportada para o problema $B$ de tal forma que $y_2^{48} > \varepsilon = 137.5$. Uma iteração simplex leva-nos à $EAG$ eficiente e suportada $EAG^{27}$ com $y_2^{27} > 137.5$. Esta solução tem $EAG^{25}$ como solução adjacente, eficiente e suportada com $y_2^{25} < 137.5$. Assim, são consideradas as $EAG$s $EAG^{(k_1)} = EAG^{25}$ e $EAG^{(k_2)} = EAG^{27}$.

Temos $\Delta_1 = 0.055556$ e $d = (0, 0, 0, 0, -1, 1, -1)$ com $x^{(k')} = (6,\ 4,\ 0,\ 6,\ 1.94444,\ 2.05556,\ 7.94444)$ e $(k, l) = (4, 5)$ é o arco de entrada permitindo passar da $EAG^{27}$ para a adjacente $EAG^{25}$. O problema $(K)$ é dividido em,

$$\begin{array}{lll} \text{“minimizar”} & (f_1(x), f_2(x)), & \\ \text{sujeito a:} & x \in X^{(K)}, & (D) \\ & x_{45} \leqslant 7, & \end{array}$$

e

$$\begin{array}{lll} \text{“minimizar”} & (f_1(x), f_2(x)), & \\ \text{sujeito a:} & x \in X^{(K)}, & (E) \\ & x_{45} = 8. & \end{array}$$

$Q = \{C, D, E\}$.

**3.** $Q \neq \{\}$, consideremos o problema $D$ (ver Figura 7.7). $Q = \{C, E\}$.

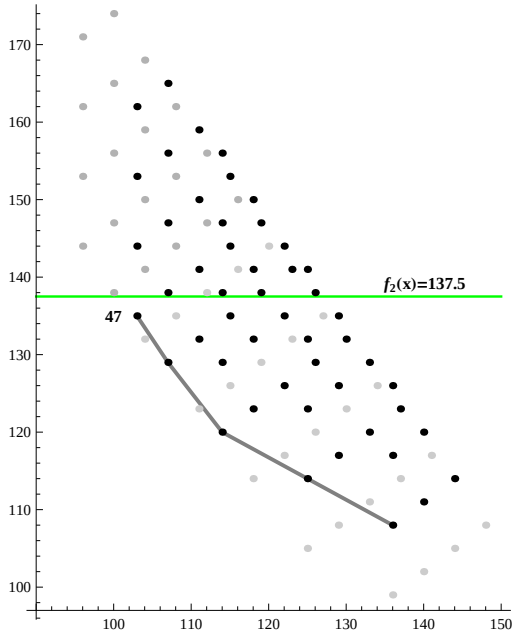



**Figura 7.7:** *Região admissível para o problema D.*

**<u>It</u> 3**

**2.** $EAG^{(k_2)}$ não existe e $EAG^{(k_1)} = EAG^{47}$. O valor $y_2^{47} = 135 < \varepsilon$ e $Apurada = (+\infty, +\infty)$, deste modo fazemos $Apurada = y^{47} = (103, 135)$.

**3.** $Q \neq \{\}$, consideremos o problema $E$ (ver Figura 7.8). $Q = \{C\}$.

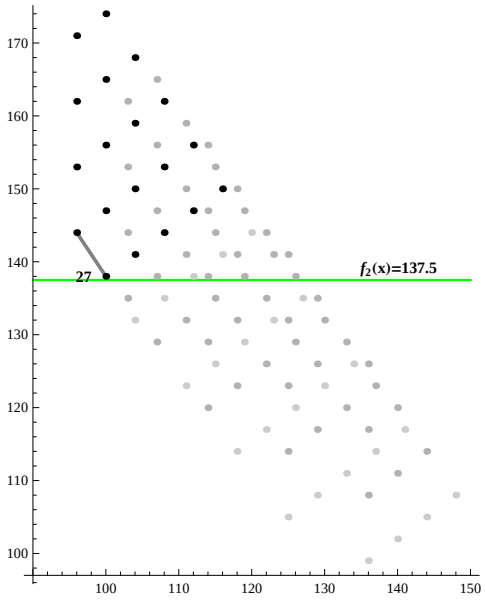



**Figura 7.8:** *Região admissível para o problema $E$.*

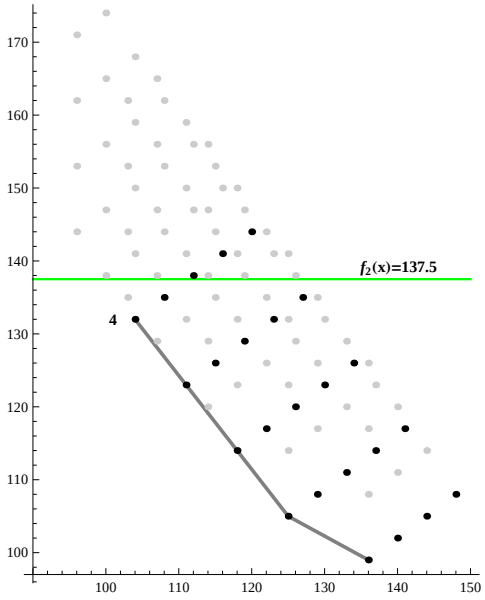



**Figura 7.9:** *Região admissível para o problema $C$.*

**It 4**

**2.** $EAG^{(k_2)}$ não existe e $EAG^{(k_1)} = EAG^{27}$. O valor $y_2^{27} = 138 > \varepsilon$ e, por isso, não actualizamos o vector *Apurada*.

**3.** $Q \neq \{\}$, consideremos o problema $C$ (ver Figura 7.9). $Q = \{\}$.

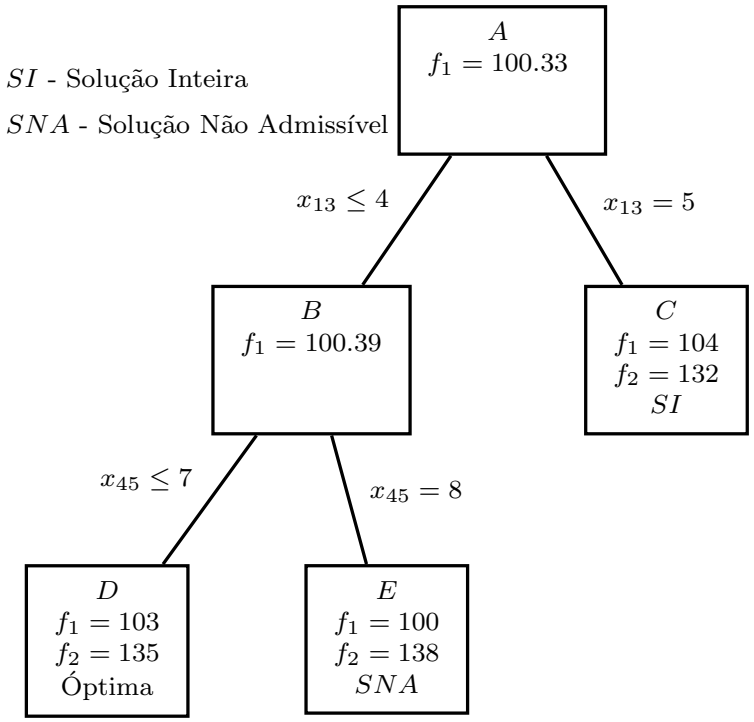



**Figura 7.10:** *Diagrama PSAP.*

**It 5**

**2.** $EAG^{(k_2)}$ não existe e $EAG^{(k_1)} = EAG^4$. O valor de $y_2^4 = 132 < \varepsilon$. $Apurada = y^{47}$ e $y_1^4 = 104 > y_1^{47} = 103$.

**3.** $Q = \{\}$, STOP. A solução não-dominada $y^{47}$ é a solução do problema actual (ver a árvore do PSAP da Figura 7.10).

O conjunto das soluções não-dominadas é actualizado e, deste modo, $ND1 = \{y^{27}, y^{47}, y^{48}\}$.

Como $y_2^{47} \neq y_2^*$ o algoritmo continua. Pára quando a solução não-dominada $y^5$ é obtida (ver Figura 7.11).

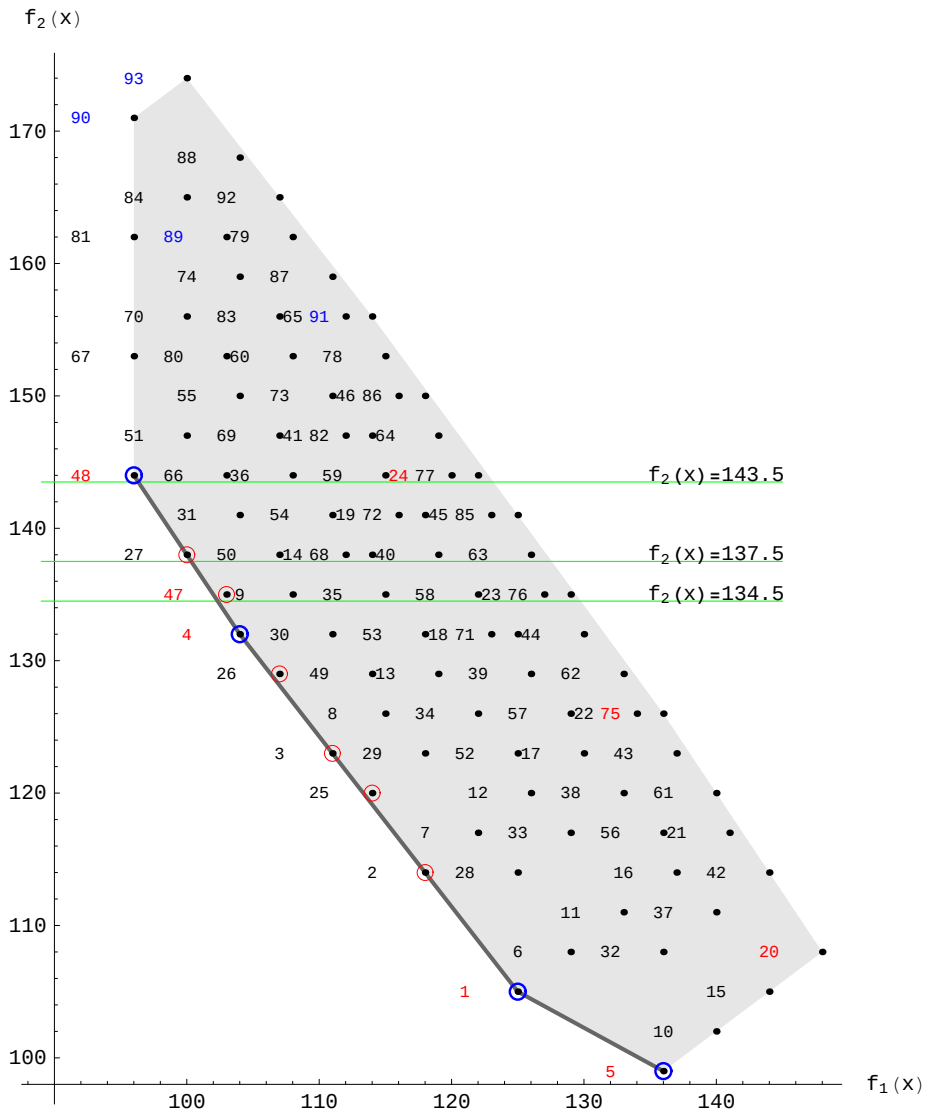



**Figura 7.11:** *Soluções não-dominadas.*

## 7.5 Resultados computacionais

O algoritmo apresentado foi implementado e executado para um grande número de instâncias do problema FRIBO (mais de 2160 instâncias). O objectivo desta secção é o de relatar os resultados para uma melhor compreensão do comportamento do algoritmo.

O algoritmo foi implementado usando linguagem de programação C. Os ensaios foram feitos num computador pessoal, equipado com um processador Intel Pentium 2.5GHz com 3GB de RAM. O programa foi executado no sistema operativo OS X.

Para cada tipo de problema foi gerado um conjunto de 30 instâncias. A escolha de 30 instâncias justifica-se com base na prática adoptada pelos estatísticos que consideram, em geral, uma amostra de tamanho inferior a 30 como uma amostra pequena para a dedução de conclusões com significado estatístico. Cada instância foi obtida usando o gerador de redes *NETGEN* (ver Klingman *et al.*, 1974), depois de algumas alterações feitas para que fosse possível gerar instâncias do problema FRIBO.

Resolvemos um conjunto de 72 tipos de problemas com 30 instâncias cada, construídas segundo o esquema da Figura 7.12, utilizando como parâmetros os usados pelo gerador *NETGEN*:

- número de nodos, $m$;

- número de arcos, $n$;

- maior valor do custo, $C$ (Os coeficientes inteiros das variáveis das duas funções objectivo são gerados aleatoriamente entre os inteiros $\{0, 1, 2, \cdots, C\}$);

- maior valor do limite máximo das capacidades dos arcos, $U$ (Todos os arcos têm capacidade limitada, sendo o limite inferior sempre igual a 0 e o superior gerado aleatoriamente entre o conjunto de inteiros no intervalo $[0, U]$);

- e oferta total, $\sum$.

De acordo com esquema da Figura 7.12, considerámos instâncias com

20 nodos e 40, 80, 120 e 156 arcos (156 arcos foi o número máximo de arcos admitido pelo *NETGEN* quando o número de nodos é igual a 20);

30 nodos e 60, 120, 180, 240 e 306 arcos (306 arcos foi o número máximo de arcos admitido pelo *NETGEN* quando o número de nodos é igual a 30).

As combinações de $U$, $C$ e $\sum$ foram escolhidas de acordo com o esquema da Figura 7.12. Assim, instâncias do tipo 1 têm $m = 20$, $n = 40$, $C = 100$, $\sum = 100$ e $U = 20$. Instâncias do tipo 2 têm os mesmos parâmetros com excepção de $U = 60$. Instâncias do tipo 3 têm agora valores $\sum = 500$ e $U = 20$. A enumeração continua desta forma até às instâncias do tipo 72 com $m = 30$, $n = 306$, $C = 1000$, $\sum = 500$ e $U = 60$.

Para cada instância, dois números exigidos pelo gerador *NETGEN*, designados por sementes, foram aleatoriamente gerados. O primeiro número no intervalo [12345678, 15345678] e o segundo no intervalo [56789123, 59789123]. Não existe qualquer razão especial para a escolha destes intervalos, outros poderiam ter sido escolhidos. De referir ainda que, apesar do número de arcos ser um dos parâmetros de entrada do gerador *NETGEN*, algumas vezes este parâmetro é modificado durante a geração de uma instância (ver Klingman *et al.*, 1974).

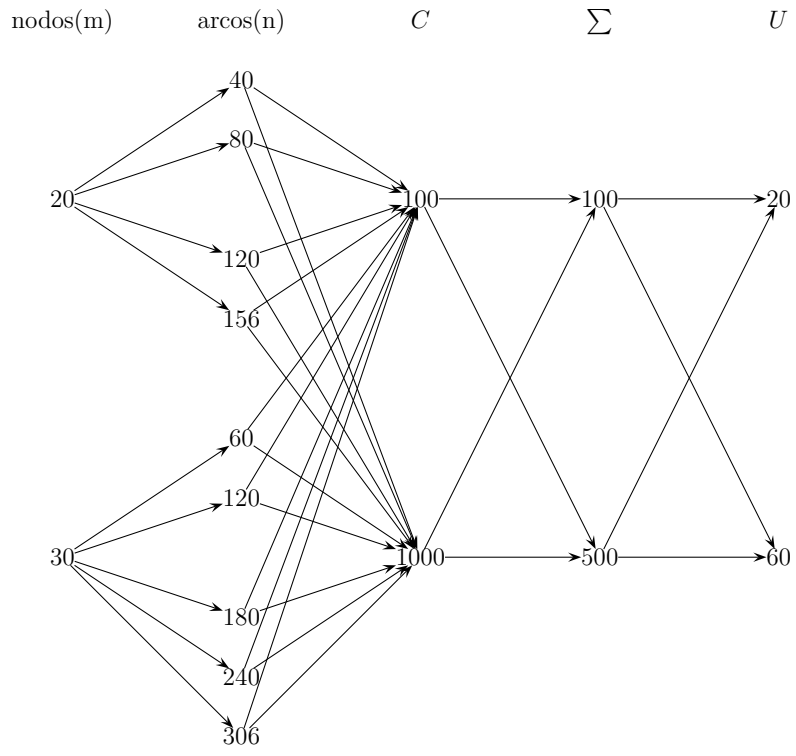



**Figura 7.12:** *Esquema para o cálculo dos parâmetros do problema.*

Os parâmetros referidos, assim como o número de soluções não-dominadas, o tempo *CPU* para determinar todas estas soluções e o número de soluções eficientes, foram registados numa base de dados. Posteriormente, fez-se um estudo estatístico

destes dados, utilizando o software SPSS versão 17.0 SPSS (2007). O estudo dos dados, obtidos através de ensaios computacionais, é abordado por Coffin & Saltzman (2000). Estes autores referem algumas das principais técnicas que devem ser utilizadas neste estudo e realçam a importância da análise estatística na aquisição de conhecimento e poder de explicação do comportamento dos algoritmos. O tamanho da amostra suficientemente grande é também essencial, com vista à aquisição de significância estatística. Na nossa análise estatística dos dados, aplicamos modelos de Regressão Linear Múltipla (RLM), com vista à explicação da variabilidade do número de soluções não-dominadas e do tempo necessário para as calcular.

O valor médio ($\bar{x}$), o mínimo (Min), o máximo (Max) e a mediana (Me) das 30 instâncias, de cada tipo, são apresentados na Tabela 7.1, tanto para o número de soluções não-dominadas como para o tempo *CPU* gasto para calcular estas soluções.

**Tabela 7.1:** *Resultados computacionais.*

| | Soluções ND | | | | Tempo *CPU* (s) | | | |
|---|---|---|---|---|---|---|---|---|
| N | $\bar{x}$ | Min | Max | Me | $\bar{x}$ | Min | Max | Me |
| 1 | 113.57 | 5 | 391 | 87.50 | 0.29 | 0.011 | 1.178 | 0.18 |
| 2 | 97.80 | 19 | 256 | 85.00 | 0.21 | 0.044 | 0.642 | 0.16 |
| 3 | 575.43 | 150 | 1 789 | 459.50 | 1.65 | 0.226 | 6.424 | 1.11 |
| 4 | 604.57 | 38 | 1 942 | 559.00 | 1.65 | 0.062 | 4.9 | 1.33 |
| 5 | 115.17 | 12 | 437 | 90.00 | 0.67 | 0.024 | 1.764 | 0.56 |
| 6 | 101.33 | 2 | 244 | 88.50 | 0.58 | 0.013 | 2.082 | 0.46 |
| 7 | 576.87 | 132 | 2 318 | 394.50 | 3.39 | 0.65 | 14.343 | 2.42 |
| 8 | 870.30 | 295 | 2 775 | 706.50 | 4.66 | 1.287 | 23.146 | 3.19 |
| 9 | 294.47 | 52 | 646 | 268.00 | 3.64 | 0.277 | 13.892 | 3.08 |
| 10 | 336.83 | 104 | 791 | 304.50 | 4.18 | 0.532 | 22.523 | 3.17 |
| 11 | 1 300.23 | 227 | 2 687 | 1 108.00 | 18.53 | 0.971 | 47.354 | 15.49 |
| 12 | 1 763.80 | 190 | 4 698 | 1 834.50 | 29.03 | 1.352 | 86.254 | 28.96 |
| 13 | 351.60 | 92 | 1 611 | 269.00 | 8.66 | 1.444 | 63.709 | 4.45 |
| 14 | 448.93 | 167 | 973 | 398.00 | 8.61 | 1.166 | 24.749 | 6.64 |
| 15 | 1 707.73 | 600 | 7 355 | 1 373.00 | 37.34 | 5.539 | 120.171 | 24.95 |
| 16 | 3 580.73 | 911 | 17 197 | 2 425.50 | 82.16 | 11.66 | 422.525 | 52.53 |
| 17 | 523.03 | 133 | 975 | 506.00 | 18.18 | 2.161 | 34.621 | 18.20 |
| 18 | 532.07 | 228 | 910 | 519.50 | 17.01 | 3.26 | 43.568 | 16.86 |
| 19 | 1 767.83 | 649 | 3 413 | 1 785.00 | 80.10 | 11.513 | 354.267 | 75.31 |
| 20 | 3 179.37 | 549 | 7 793 | 2 873.00 | 134.40 | 6.033 | 286.355 | 126.93 |
| 21 | 615.03 | 241 | 1 347 | 576.00 | 30.36 | 7.503 | 72.652 | 27.17 |
| 22 | 702.53 | 292 | 1 693 | 611.50 | 31.18 | 6.913 | 80.043 | 24.83 |
| 23 | 3 024.07 | 882 | 6 687 | 2 740.50 | 175.28 | 29.786 | 428.957 | 147.74 |

*continua na próxima página*

*continuação da página anterior*

| | Soluções ND | | | | Tempo *CPU* (s) | | | |
|---|---|---|---|---|---|---|---|---|
| N | $\bar{x}$ | Min | Max | Me | $\bar{x}$ | Min | Max | Me |
| 24 | 4 178.90 | 1 512 | 9 231 | 3 695.00 | 239.18 | 43.543 | 713.566 | 215.07 |
| 25 | 764.70 | 423 | 1 544 | 735.00 | 61.93 | 20.242 | 127.17 | 55.47 |
| 26 | 754.50 | 397 | 1 554 | 704.50 | 50.35 | 20.617 | 104.27 | 46.34 |
| 27 | 2 983.47 | 1 240 | 5 914 | 3 077.00 | 333.05 | 65.309 | 1 161.936 | 283.10 |
| 28 | 4 205.43 | 2 104 | 6 853 | 4 189.50 | 480.41 | 125.621 | 1 186.836 | 440.43 |
| 29 | 855.23 | 322 | 2 061 | 813.00 | 80.20 | 23.48 | 205.406 | 73.54 |
| 30 | 885.70 | 279 | 1 774 | 875.00 | 80.83 | 21.375 | 182.449 | 75.88 |
| 31 | 4 617.73 | 1 874 | 10 112 | 4 181.50 | 625.66 | 153.453 | 2 456.594 | 503.18 |
| 32 | 5 817.00 | 2 336 | 13 229 | 5 475.00 | 673.65 | 144.211 | 2 312.055 | 630.70 |
| 33 | 91.27 | 25 | 325 | 78.50 | 0.39 | 0.053 | 1.861 | 0.31 |
| 34 | 93.10 | 19 | 248 | 80.50 | 0.44 | 0.042 | 1.663 | 0.37 |
| 35 | 502.70 | 63 | 1 679 | 386.50 | 2.98 | 0.121 | 28.739 | 1.63 |
| 36 | 747.37 | 92 | 3 485 | 656.00 | 4.02 | 0.204 | 23.831 | 2.71 |
| 37 | 107.63 | 9 | 331 | 81.50 | 0.89 | 0.036 | 3.83 | 0.69 |
| 38 | 147.83 | 42 | 565 | 113.50 | 1.27 | 0.423 | 3.648 | 0.97 |
| 39 | 573.90 | 32 | 1 870 | 485.50 | 4.59 | 0.242 | 14.244 | 4.07 |
| 40 | 614.27 | 24 | 1 480 | 521.00 | 4.84 | 0.208 | 16.537 | 3.85 |
| 41 | 36.00 | 13 | 102 | 27.00 | 0.66 | 0.119 | 4.38 | 0.45 |
| 42 | 338.30 | 27 | 780 | 329.00 | 11.65 | 0.22 | 38.808 | 9.25 |
| 43 | 1 675.30 | 362 | 3 401 | 1 651.50 | 71.90 | 6.869 | 162.427 | 55.18 |
| 44 | 2 207.47 | 925 | 4 910 | 1 940.00 | 98.76 | 12.037 | 494.366 | 69.69 |
| 45 | 403.70 | 150 | 976 | 342.00 | 20.20 | 3.309 | 54.819 | 17.75 |
| 46 | 386.77 | 150 | 696 | 371.00 | 19.55 | 4.258 | 44.236 | 17.73 |
| 47 | 2 297.20 | 681 | 5 100 | 2 066.00 | 130.22 | 18.566 | 326.761 | 103.84 |
| 48 | 3 597.90 | 1 024 | 8 888 | 3 300.50 | 202.30 | 36.978 | 707.646 | 139.90 |
| 49 | 542.17 | 206 | 953 | 541.50 | 64.49 | 11.656 | 140.511 | 58.76 |
| 50 | 571.13 | 221 | 992 | 588.00 | 61.32 | 15.391 | 129.027 | 62.16 |
| 51 | 3 234.20 | 1 617 | 6 183 | 2 899.00 | 539.51 | 170.234 | 1 386.575 | 468.07 |
| 52 | 3 836.90 | 2 048 | 8 772 | 3 608.50 | 550.64 | 181.945 | 1 805.395 | 439.72 |
| 53 | 613.43 | 185 | 1 294 | 573.00 | 87.67 | 16.441 | 174.184 | 81.46 |
| 54 | 679.43 | 373 | 1 287 | 648.50 | 108.48 | 34.969 | 269.078 | 98.78 |
| 55 | 3 929.47 | 2 158 | 7 510 | 3 423.00 | 799.14 | 284.926 | 2 000.449 | 615.69 |
| 56 | 4 858.10 | 1 852 | 9 425 | 4 483.00 | 878.01 | 163.641 | 2 547.984 | 798.18 |
| 57 | 719.60 | 307 | 1 126 | 677.50 | 193.94 | 74.234 | 358.977 | 173.66 |
| 58 | 733.07 | 403 | 1 104 | 695.50 | 194.02 | 80.484 | 353.578 | 186.43 |
| 59 | 4 563.30 | 2 486 | 6 643 | 4 695.00 | 1 892.74 | 505.879 | 3 388.99 | 1 911.86 |
| 60 | 4 643.60 | 2 310 | 9 170 | 4 859.00 | 1 587.67 | 589.191 | 3 375.016 | 1 537.61 |
| 61 | 867.97 | 445 | 1 502 | 887.50 | 287.46 | 85.836 | 549.68 | 278.50 |
| 62 | 897.50 | 416 | 1 544 | 862.00 | 298.17 | 116.031 | 726.563 | 274.47 |
| 63 | 6 386.53 | 3 797 | 10 619 | 6 304.50 | 3 092.06 | 1 287.125 | 5 970.688 | 2 762.59 |
| 64 | 7 568.33 | 4 506 | 13 667 | 7 083.50 | 3 046.85 | 948.422 | 9 036.875 | 2 625.93 |

*continua na próxima página*

*continuação da página anterior*

| | Soluções ND | | | | Tempo *CPU* (s) | | | |
|---|---|---|---|---|---|---|---|---|
| N | $\bar{x}$ | Min | Max | Me | $\bar{x}$ | Min | Max | Me |
| 65 | 928.70 | 503 | 1 348 | 977.50 | 513.91 | 195.721 | 1 263.803 | 549.12 |
| 66 | 974.50 | 523 | 1 618 | 979.00 | 509.94 | 248.315 | 1 127.118 | 485.96 |
| 67 | 5 354.77 | 3 120 | 8 007 | 5 502.50 | 4 528.82 | 2 135.723 | 9 163.605 | 4 441.91 |
| 68 | 5 560.97 | 3 324 | 8 349 | 5 262.50 | 3 629.21 | 1 369.25 | 7 638.859 | 3 412.84 |
| 69 | 1 216.00 | 663 | 1 798 | 1 189.00 | 801.09 | 362.969 | 1 205.906 | 807.34 |
| 70 | 1 246.93 | 565 | 1 973 | 1 172.00 | 831.46 | 260.125 | 1 551.906 | 683.16 |
| 71 | 8 629.67 | 3 819 | 16 374 | 8 370.50 | 8 566.01 | 2 786.063 | 20 199.67 | 8 127.72 |
| 72 | 10 070.43 | 5 117 | 17 127 | 9 983.00 | 7 688.51 | 2 994.73 | 15 212.34 | 6 925.04 |

As 30 instâncias mais difíceis, isto é, as que gastaram mais tempo *CPU* no cálculo das soluções não-dominadas têm valor de $\sum = 500$; um número de nodos igual a 30; apenas uma destas instâncias tem um número de arcos igual a 240, todas as restantes têm o maior número de arcos utilizado, que foi 306; 18 instâncias têm $U = 20$ e 12 têm $U = 60$; apenas uma delas tem $C = 100$, tendo as restantes $C = 1000$. O tempo *CPU* utilizado pelo algoritmo depende do número de soluções não-dominadas, uma vez que o cálculo de uma destas soluções é feito resolvendo um problema de restrição-$\varepsilon$. A determinação da solução óptima deste problema depende da árvore do PSAP. Esta será a razão para um maior ou menor tempo de *CPU* gasto no cálculo de todas as soluções não-dominadas.

De seguida, descrevemos alguns dos resultados estatísticos obtidos a partir dos dados recolhidos. O número médio de soluções não-dominadas e o tempo *CPU* médio para encontrar todas estas soluções, para cada instância, cresce quando o limite superior do intervalo onde se geram os custos passa de 100 para 1000. O mesmo acontece quando a oferta total passa de 100 para 500. Comparando instâncias obtidas utilizando o parâmetro $C = 1000$ com as que têm $C = 100$, as primeiras têm, em média, mais 733.07 soluções não-dominadas do que as segundas. E o tempo *CPU* necessário para resolver estas instâncias (calcular todas as soluções não-dominadas) é, em média, de mais 368.32 segundos que o gasto pelas instâncias com $C = 100$. Em média, instâncias com $\sum = 500$ têm mais 995.42 soluções não-dominadas que as instâncias com $\sum = 100$. E o tempo *CPU* necessário para resolver estas instâncias é maior (mais 1395.74 segundos, em média) que o tempo *CPU* para resolver as instâncias com $\sum = 100$. As instâncias com $U = 60$ têm, em média mais 415.95 soluções não-dominadas e gastam menos 42.01 segundos que as instâncias com $U = 20$. Em média, as instâncias consideradas têm mais 69.62 soluções eficientes que soluções não-dominadas. O valor mínimo para esta diferença é 0 e o valor máximo é 6092.

Consideremos o modelo de regressão múltipla (uma das técnicas apresentadas

em Coffin & Saltzman, 2000),

$$snd = b_0 + b_1 \sum + b_2 n + b_3 C + b_4 m + b_5 U + \xi \tag{7.4}$$

onde, as variáveis *snd*, $\sum$, $n$, $C$, $m$ e $U$ representam o número de soluções não--dominadas, a oferta total disponível na rede, o número de arcos, o limite superior dos custos, o número de nodos e a capacidade máxima de cada arco na rede, respectivamente, e $\xi$ é um termo aleatório. O modelo de regressão ajustado, com os coeficientes de regressão arredondados na terceira casa decimal, é o seguinte,

$$snd \approx -2042.176 + 7.120 \sum + 16.400 n + 0.815 C - 54.066 m + 10.401 U.$$

Os coeficientes estandardizados são 0.580, 0.543, 0.149, $-0.109$ e 0.085 para $b_1$, $b_2$, $b_3$, $b_4$ e $b_5$, respectivamente. O coeficiente de determinação múltipla ajustado ($R^2$ ajustado) é 0.613. Assim, podemos dizer que 61.3% da variabilidade de *snd* é explicada pelas variáveis $\sum$, $n$, $C$, $m$ e $U$. Podemos também dizer que $\sum$ e $n$ são as variáveis que originam a maior alteração em *snd*. O modelo correspondente, com as variáveis $\sum$ e $n$ apenas, explica 57.4% da variabilidade de *snd*. Podemos dizer que a fraca contribuição de $m$ para a variabilidade do número de soluções não-dominadas é negativa, isto é, nos nossos dados (resultados) um valor maior de $m$ conduz a um menor valor do número de soluções não-dominadas.

Consideremos, agora, o modelo de regressão múltipla para o tempo *CPU*.

$$tempo = b_0 + b_1 \sum + b_2 n + b_3 C + b_4 m + b_5 U + \xi \tag{7.5}$$

onde, *tempo* é o tempo *CPU*. O procedimento estatístico por passos (*stepwise*) apaga os termos $b_4 m$ e $b_5 U$ do modelo de regressão ajustado. Isto significa que existe evidência estatística para considerarmos $b_4$ e $b_5$ nulos. O modelo de regressão ajustado é então

$$tempo \approx -2011.512 + 2.489 \sum + 11.447 n + 0.410 C.$$

O coeficiente de determinação múltipla ajustado é 0.406 e os coeficientes estandardizados são 0.296, 0.554 e 0.110 para $b_1$, $b_2$ e $b_3$, respectivamente. Podemos, por isso, dizer que 40.6% da variabilidade do tempo *CPU* é explicada pelas variáveis $\sum$, $n$ e $C$. Além disso, a variável que origina a maior variabilidade do tempo é o número de arcos $n$.

Considerando a variável dependente como o tempo *CPU* e a independente como o número de soluções não-dominadas *snd* (ver Figura 7.13), podemos dizer que 60.1% da variabilidade do tempo *CPU* é devida ao número de soluções não-dominadas.

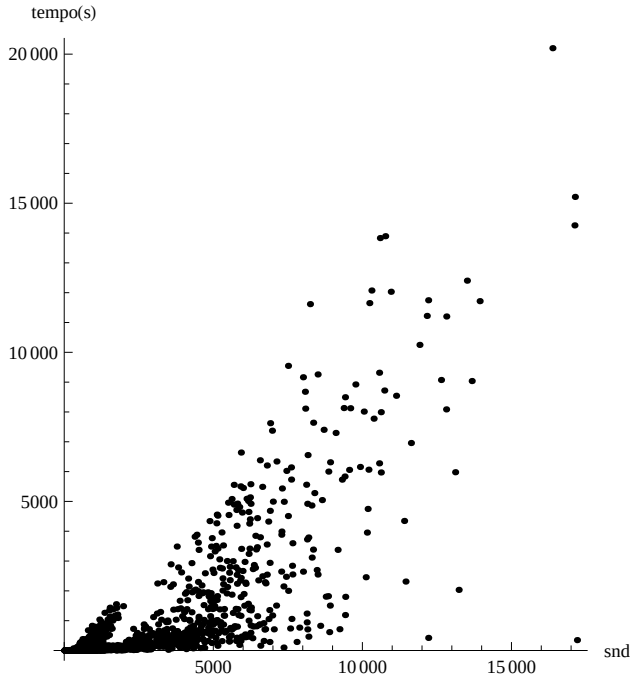



**Figura 7.13:** *Diagrama de dispersão do número de soluções não-dominadas versus tempo CPU.*

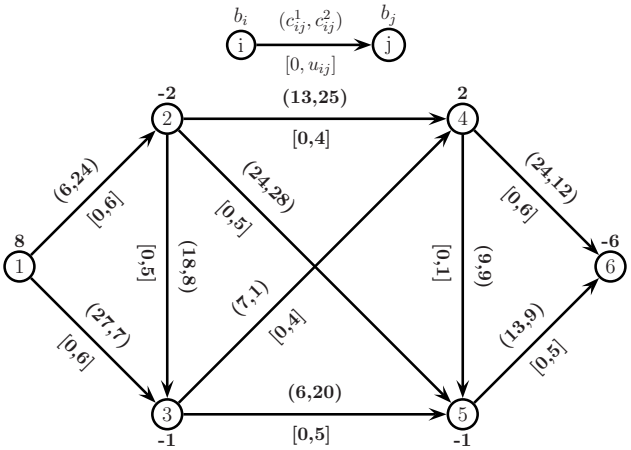



**Figura 7.14:** *Rede de um problema FRBO.*

Usando os dados reunidos podemos afirmar que a variabilidade do tempo *CPU*

gasto para calcular todas as soluções não-dominadas é devida, principalmente, ao número de soluções não-dominadas. A maior parte da variabilidade deste número é explicada pela oferta total disponível na rede, o número de arcos e o custo máximo de cada unidade de fluxo em cada arco.

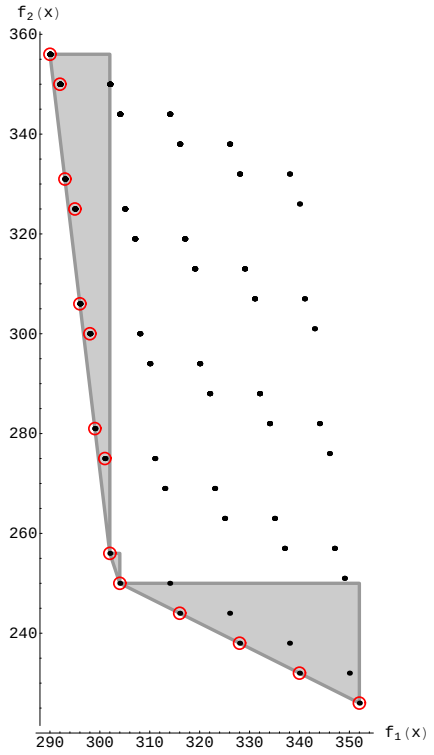



**Figura 7.15:** *Soluções não-dominadas.*

O algoritmo pode calcular também todas as soluções eficientes. Como exemplo, o programa foi executado para o exemplo da Figura 7.14. Um conjunto de 14 soluções não-dominadas: $\{(290, 356), (292, 350), (293, 331), (295, 325), (296, 306), (298, 300), (299, 281), (301, 275), (302, 256), (304, 250), (316, 244), (328, 238), (340, 232), (352, 226)\}$ (ver Figura 7.15) corresponde a um conjunto de 74 soluções eficientes (ver Tabela 7.2). Para obter todas as soluções eficientes é necessário um maior tempo *CPU*. Para percebermos a forma como o algoritmo funciona consideremos a resolução do problema de restrição $\varepsilon$ com $\varepsilon = 280.5$. A árvore do PSAP é apresentada na Figura 7.16. Existem três soluções eficientes nos nodos 10, 18 e 28. Se pretendêssemos calcular apenas as soluções não-dominadas, os ramos da árvore abaixo dos nodos 14 e 22 poderiam ser cortados, nestes nodos, com $f_1 = 300.34$. Qualquer solução

nestes ramos teria $f_1 \geqslant 301$ e $f_2 \geqslant 275$. Era, por isso, evitado continuar a explorar estes ramos procurando uma solução melhor. Neste caso, como pretendemos todas as soluções eficientes, não podemos abandonar estes ramos.

**Tabela 7.2:** *Soluções eficientes do problema da Figura 7.14.*

| Nº | $x_{12}$ | $x_{13}$ | $x_{23}$ | $x_{24}$ | $x_{25}$ | $x_{34}$ | $x_{35}$ | $x_{45}$ | $x_{46}$ | $x_{56}$ | $y_1$ | $y_2$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 6 | 2 | 0 | 0 | 4 | 0 | 1 | 1 | 1 | 5 | 290 | 356 |
| 2 | 6 | 2 | 0 | 1 | 3 | 0 | 1 | 1 | 2 | 4 | 290 | 356 |
| 3 | 6 | 2 | 0 | 2 | 2 | 0 | 1 | 1 | 3 | 3 | 290 | 356 |
| 4 | 6 | 2 | 0 | 3 | 1 | 0 | 1 | 1 | 4 | 2 | 290 | 356 |
| 5 | 6 | 2 | 0 | 4 | 0 | 0 | 1 | 1 | 5 | 1 | 290 | 356 |
| 6 | 6 | 2 | 1 | 0 | 3 | 0 | 2 | 1 | 1 | 5 | 290 | 356 |
| 7 | 6 | 2 | 1 | 1 | 2 | 0 | 2 | 1 | 2 | 4 | 290 | 356 |
| 8 | 6 | 2 | 1 | 2 | 1 | 0 | 2 | 1 | 3 | 3 | 290 | 356 |
| 9 | 6 | 2 | 1 | 3 | 0 | 0 | 2 | 1 | 4 | 2 | 290 | 356 |
| 10 | 6 | 2 | 2 | 0 | 2 | 0 | 3 | 1 | 1 | 5 | 290 | 356 |
| 11 | 6 | 2 | 2 | 1 | 1 | 0 | 3 | 1 | 2 | 4 | 290 | 356 |
| 12 | 6 | 2 | 2 | 2 | 0 | 0 | 3 | 1 | 3 | 3 | 290 | 356 |
| 13 | 6 | 2 | 3 | 0 | 1 | 0 | 4 | 1 | 1 | 5 | 290 | 356 |
| 14 | 6 | 2 | 3 | 1 | 0 | 0 | 4 | 1 | 2 | 4 | 290 | 356 |
| 15 | 6 | 2 | 4 | 0 | 0 | 0 | 5 | 1 | 1 | 5 | 290 | 356 |
| 16 | 6 | 2 | 0 | 0 | 4 | 0 | 1 | 0 | 2 | 4 | 292 | 350 |
| 17 | 6 | 2 | 0 | 1 | 3 | 0 | 1 | 0 | 3 | 3 | 292 | 350 |
| 18 | 6 | 2 | 0 | 2 | 2 | 0 | 1 | 0 | 4 | 2 | 292 | 350 |
| 19 | 6 | 2 | 0 | 3 | 1 | 0 | 1 | 0 | 5 | 1 | 292 | 350 |
| 20 | 6 | 2 | 0 | 4 | 0 | 0 | 1 | 0 | 6 | 0 | 292 | 350 |
| 21 | 6 | 2 | 1 | 0 | 3 | 0 | 2 | 0 | 2 | 4 | 292 | 350 |
| 22 | 6 | 2 | 1 | 1 | 2 | 0 | 2 | 0 | 3 | 3 | 292 | 350 |
| 23 | 6 | 2 | 1 | 2 | 1 | 0 | 2 | 0 | 4 | 2 | 292 | 350 |
| 24 | 6 | 2 | 1 | 3 | 0 | 0 | 2 | 0 | 5 | 1 | 292 | 350 |
| 25 | 6 | 2 | 2 | 0 | 2 | 0 | 3 | 0 | 2 | 4 | 292 | 350 |
| 26 | 6 | 2 | 2 | 1 | 1 | 0 | 3 | 0 | 3 | 3 | 292 | 350 |
| 27 | 6 | 2 | 2 | 2 | 0 | 0 | 3 | 0 | 4 | 2 | 292 | 350 |
| 28 | 6 | 2 | 3 | 0 | 1 | 0 | 4 | 0 | 2 | 4 | 292 | 350 |
| 29 | 6 | 2 | 3 | 1 | 0 | 0 | 4 | 0 | 3 | 3 | 292 | 350 |
| 30 | 6 | 2 | 4 | 0 | 0 | 0 | 5 | 0 | 2 | 4 | 292 | 350 |
| 31 | 5 | 3 | 0 | 3 | 0 | 0 | 2 | 1 | 4 | 2 | 293 | 331 |
| 32 | 5 | 3 | 0 | 2 | 1 | 0 | 2 | 1 | 3 | 3 | 293 | 331 |
| 33 | 5 | 3 | 0 | 1 | 2 | 0 | 2 | 1 | 2 | 4 | 293 | 331 |
| 34 | 5 | 3 | 0 | 0 | 3 | 0 | 2 | 1 | 1 | 5 | 293 | 331 |

*continua na próxima página*

*continuação da página anterior*

| Nº | $x_{12}$ | $x_{13}$ | $x_{23}$ | $x_{24}$ | $x_{25}$ | $x_{34}$ | $x_{35}$ | $x_{45}$ | $x_{46}$ | $x_{56}$ | $y_1$ | $y_2$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 35 | 5 | 3 | 1 | 2 | 0 | 0 | 3 | 1 | 3 | 3 | 293 | 331 |
| 36 | 5 | 3 | 1 | 1 | 1 | 0 | 3 | 1 | 2 | 4 | 293 | 331 |
| 37 | 5 | 3 | 1 | 0 | 2 | 0 | 3 | 1 | 1 | 5 | 293 | 331 |
| 38 | 5 | 3 | 2 | 1 | 0 | 0 | 4 | 1 | 2 | 4 | 293 | 331 |
| 39 | 5 | 3 | 2 | 0 | 1 | 0 | 4 | 1 | 1 | 5 | 293 | 331 |
| 40 | 5 | 3 | 3 | 0 | 0 | 0 | 5 | 1 | 1 | 5 | 293 | 331 |
| 41 | 5 | 3 | 0 | 0 | 3 | 0 | 2 | 0 | 2 | 4 | 295 | 325 |
| 42 | 5 | 3 | 0 | 1 | 2 | 0 | 2 | 0 | 3 | 3 | 295 | 325 |
| 43 | 5 | 3 | 0 | 2 | 1 | 0 | 2 | 0 | 4 | 2 | 295 | 325 |
| 44 | 5 | 3 | 0 | 3 | 0 | 0 | 2 | 0 | 5 | 1 | 295 | 325 |
| 45 | 5 | 3 | 1 | 0 | 2 | 0 | 3 | 0 | 2 | 4 | 295 | 325 |
| 46 | 5 | 3 | 1 | 1 | 1 | 0 | 3 | 0 | 3 | 3 | 295 | 325 |
| 47 | 5 | 3 | 1 | 2 | 0 | 0 | 3 | 0 | 4 | 2 | 295 | 325 |
| 48 | 5 | 3 | 2 | 0 | 1 | 0 | 4 | 0 | 2 | 4 | 295 | 325 |
| 49 | 5 | 3 | 2 | 1 | 0 | 0 | 4 | 0 | 3 | 3 | 295 | 325 |
| 50 | 5 | 3 | 3 | 0 | 0 | 0 | 5 | 0 | 2 | 4 | 295 | 325 |
| 51 | 4 | 4 | 0 | 2 | 0 | 0 | 3 | 1 | 3 | 3 | 296 | 306 |
| 52 | 4 | 4 | 0 | 1 | 1 | 0 | 3 | 1 | 2 | 4 | 296 | 306 |
| 53 | 4 | 4 | 0 | 0 | 2 | 0 | 3 | 1 | 1 | 5 | 296 | 306 |
| 54 | 4 | 4 | 1 | 1 | 0 | 0 | 4 | 1 | 2 | 4 | 296 | 306 |
| 55 | 4 | 4 | 1 | 0 | 1 | 0 | 4 | 1 | 1 | 5 | 296 | 306 |
| 56 | 4 | 4 | 2 | 0 | 0 | 0 | 5 | 1 | 1 | 5 | 296 | 306 |
| 57 | 4 | 4 | 0 | 2 | 0 | 0 | 3 | 0 | 4 | 2 | 298 | 300 |
| 58 | 4 | 4 | 0 | 1 | 1 | 0 | 3 | 0 | 3 | 3 | 298 | 300 |
| 59 | 4 | 4 | 0 | 0 | 2 | 0 | 3 | 0 | 2 | 4 | 298 | 300 |
| 60 | 4 | 4 | 1 | 1 | 0 | 0 | 4 | 0 | 3 | 3 | 298 | 300 |
| 61 | 4 | 4 | 1 | 0 | 1 | 0 | 4 | 0 | 2 | 4 | 298 | 300 |
| 62 | 4 | 4 | 2 | 0 | 0 | 0 | 5 | 0 | 2 | 4 | 298 | 300 |
| 63 | 3 | 5 | 0 | 0 | 1 | 0 | 4 | 1 | 1 | 5 | 299 | 281 |
| 64 | 3 | 5 | 0 | 1 | 0 | 0 | 4 | 1 | 2 | 4 | 299 | 281 |
| 65 | 3 | 5 | 1 | 0 | 0 | 0 | 5 | 1 | 1 | 5 | 299 | 281 |
| 66 | 3 | 5 | 0 | 0 | 1 | 0 | 4 | 0 | 2 | 4 | 301 | 275 |
| 67 | 3 | 5 | 0 | 1 | 0 | 0 | 4 | 0 | 3 | 3 | 301 | 275 |
| 68 | 3 | 5 | 1 | 0 | 0 | 0 | 5 | 0 | 2 | 4 | 301 | 275 |
| 69 | 2 | 6 | 0 | 0 | 0 | 0 | 5 | 1 | 1 | 5 | 302 | 256 |
| 70 | 2 | 6 | 0 | 0 | 0 | 0 | 5 | 0 | 2 | 4 | 304 | 250 |
| 71 | 2 | 6 | 0 | 0 | 0 | 1 | 4 | 0 | 3 | 3 | 316 | 244 |
| 72 | 2 | 6 | 0 | 0 | 0 | 2 | 3 | 0 | 4 | 2 | 328 | 238 |
| 73 | 2 | 6 | 0 | 0 | 0 | 3 | 2 | 0 | 5 | 1 | 340 | 232 |
| 74 | 2 | 6 | 0 | 0 | 0 | 4 | 1 | 0 | 6 | 0 | 352 | 226 |

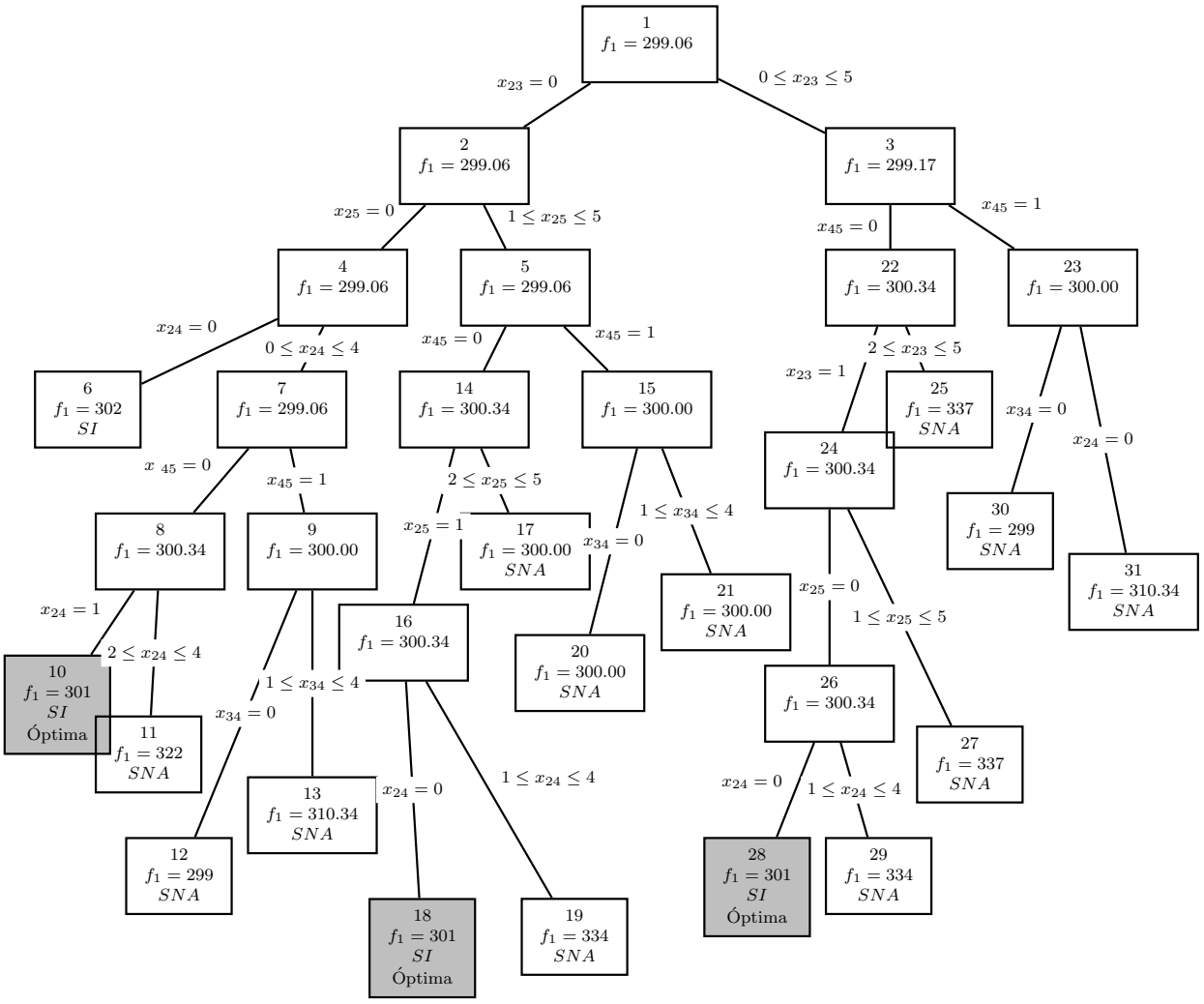



**Figura 7.16:** *Exemplo da árvore PSAP no cálculo de todas as soluções eficientes.*

# Capítulo 8

# Representação do conjunto das soluções não-dominadas[1]

Neste capítulo apresentamos o cálculo de uma "boa" representação do conjunto de todas as soluções não-dominadas para o problema FRIBO. Fazemos uma revisão sobre o que se entende por "boa" representação e introduzimos os conceitos e definições necessários. Descrevemos um novo algoritmo para o cálculo da representação referida da seguinte forma: em primeiro lugar, propomos a versão geral do algoritmo, que pode ser adaptada a diversos tipos de representações; seguidamente descrevemos adaptações do algoritmo ao cálculo de dois tipos de representação particulares. Apresentamos um exemplo ilustrativo sobre o cálculo de uma representação. Finalmente, relatamos as experiências computacionais, listamos os resultados e fazemos uma análise dos resultados.

[1] O trabalho deste capítulo, nomeadamente no que concerne ao algoritmos dos trapézios, resultou de uma conversa com M. Ehrgott, H. Hamacher e S. Ruzika. Provavelmente alguns desenvolvimentos que se seguirão a partir da conclusão desta Tese serão feitos em colaboração com estes investigadores.

## 8.1 Representação

De seguida fazemos a revisão de alguns conceitos sobre uma "boa" representação do conjunto das soluções não-dominadas em programação matemática multi-objectivo. Estamos interessados na obtenção de uma representação, em vez do conjunto de todas as soluções não-dominadas. A necessidade de termos apenas uma parte deste conjunto deve-se, fundamentalmente, ao número elevado de soluções não-dominadas, para o problema de fluxos em redes multi-objectivo. Uma *representação* de um conjunto de elementos é um conjunto, em geral, com um número muito menor de elementos, para representar o primeiro. Esses elementos podem ou não ser elementos desse conjunto, correspondendo a dois tipos de representação diferentes. Dizemos que os elementos do segundo conjunto representam os elementos do primeiro e por isso são designados por *elementos representativos*. A representação que abordamos neste capítulo é do primeiro tipo, isto é, os seus elementos são soluções não-dominadas. Os elementos da representação deverão ser escolhidos de forma a conservar as características dos elementos que representam. Algumas regras, para a escolha da representação do conjunto das soluções não-dominadas, para um problema de programação multi-objectivo, podem ser encontradas na literatura (ver Ruzika & Wiecek, 2005, Sayin, 1999). Entre estas regras referem-se três dimensões de interesse: Cobertura, Uniformidade e Cardinalidade.

1. Um conjunto diz-se uma *cobertura* se contém pontos de todas as regiões do conjunto das soluções não-dominadas sem que falhe alguma.

2. Uma representação deve ser *uniforme* no sentido de que não deve incluir mais do que um ponto de uma mesma região do conjunto das soluções não--dominadas.

3. O número de elementos, *cardinalidade*, da representação não pode ser demasiado elevado, pois aumentaria a dificuldade na sua interpretação e também a necessidade de recursos computacionais como a memória e tempo *CPU*. Por outro lado, um número demasiadamente pequeno de elementos, na representação, pode deixar por explorar algumas regiões com características especiais.

Usaremos três medidas da qualidade da representação, para medir as três dimensões referidas acima na representação $Y_r$ de $ND(Y^I)$, como em Sayin (1999):

1. $\gamma$ como o valor máximo da distância de qualquer solução não-dominada para a solução mais próxima na representação;

2. $\delta$ como o valor mínimo da distância de qualquer solução na representação para a solução mais próxima nessa mesma representação; e

3. $|Y_r|$ como o número de soluções do conjunto $Y_r$.

**Definição 8.1.1 (Representação $\gamma$)** *Consideremos um número real $\gamma > 0$ e um subconjunto não vazio $Y_r \subseteq ND(Y^I)$. Diz-se que o conjunto $Y_r$ é uma representação $\gamma$ de $ND(Y^I)$ se e só se para todo $y \in ND(Y^I)$ existir um $y_r \in Y_r$ tal que $d(y, y_r) \leqslant \gamma$, onde $d(y, y_r)$ representa a distância entre os pontos $y$ e $y_r$.*

**Definição 8.1.2 (Representação uniforme $\delta$)** *Consideremos um número real $\delta > 0$ e um subconjunto não vazio $Y_r \subseteq ND(Y^I)$. Diz-se que $Y_r$ é uma representação uniforme $\delta$ de $ND(Y^I)$ se e só se para todo $y_r^{'}, y_r^{''} \in Y_r$, $y_r^{'} \neq y_r^{''}$, $d(y_r^{'}, y_r^{''}) \geqslant \delta$.*

Definimos dois tipos de representação, mas outras representações poderiam ser definidas. Por exemplo, uma representação com um determinado número de elementos fixos. Para as representações definidas estamos interessados em representações com um pequeno valor de $\gamma$, um pequeno valor de $|Y_r|$ e um grande valor de $\delta$. Estes valores estão relacionados e, por exemplo, quando o valor de $\gamma$ diminui o valor de $|Y_r|$ cresce; quanto maior é o valor de $\delta$ maior é o valor de $\gamma$. Desta forma, não podemos melhorar todas as medidas da qualidade de uma representação em simultâneo e teremos que encontrar um compromisso entre elas.

## 8.2 Algoritmo aproximativo segundo a técnica dos trapézios

Nesta secção apresentamos um algoritmo que determina representações do conjunto das soluções não-dominadas do problema FRIBO. Este algoritmo utiliza o algoritmo de restrição-$\varepsilon$ apresentado no Capítulo 7, para determinar soluções não-dominadas para o problema FRIBO. Como vimos o algoritmo de restrição-$\varepsilon$ consegue calcular qualquer solução não-dominada do problema FRIBO. Desta forma, definindo quais as propriedades do tipo de soluções não-dominadas que pretendemos, podemos utilizá-lo para obter um conjunto aproximado de soluções não-dominadas para este problema. O procedimento poderá ser integrado num sistema interactivo de apoio ao utilizador. O algoritmo proposto poderá ser utilizado para determinar vários tipos de representação sendo preciso que o utilizador em co-interacção com o analista opte pelo tipo de representação pretendida ou, possivelmente, por determinar vários tipos de representação fazendo depois a sua comparação.

Na Secção 8.1 definimos as medidas para a avaliação de uma representação do conjunto de soluções não-dominadas. Essas medidas são conflituosas, isto é, a melhoria de uma das medidas implica, geralmente, a degradação de outra. Em particular, se pretendermos um valor de $\gamma$ grande, não podemos esperar um valor de $\delta$ pequeno e *vice-versa*, pois quando um dos valores aumenta o outro também aumenta e quando

um diminui o outro também diminui. Deste modo não é possível falar de uma "boa" representação, mas em "boas" representações alternativas de acordo com as escolhas dos diversos parâmetros. A seguir apresentamos a forma geral do algoritmo que pode ser modificado de acordo com as especificações pretendidas, relativamente à representação desejada.

### 8.2.1 Apresentação do algoritmo

O algoritmo determina uma representação do conjunto das soluções não-dominadas do problema FRIBO. O utilizador e o analista devem à partida escolher um de entre os vários tipos de representações que o algoritmo pode determinar. Vários critérios podem ser tidos em conta. Entre eles temos uma representação $\gamma$, uma representação uniforme $\delta$, um determinado número de elementos na representação ou qualquer combinação destas. O utilizador pode estar limitado no tempo e por isso interessado numa "boa" representação, utilizando o tempo disponível para o seu cálculo.

*Algoritmo 9*: Algoritmo dos trapézios.
{*Determina uma representação do conjunto das soluções não-dominadas do problema FRIBO.* }

---

**input:** Rede $\mathcal{N} = (\mathcal{G}, c, l, u, b)$.
**output:** Representação: $Y_r$.

(1) **begin**
(2) Determinar o conjunto inicial $Y_r$;
(3) $k \leftarrow 1$;
(4) **while** (não se verificar a condição de paragem) **do**
(5) **begin**
(6) Determinar o valor de $\varepsilon$;
(7) Calcular a solução óptima $y^{(k+1)}$ do problema de restrição-$\varepsilon$;
(8) $Y_r \leftarrow Y_r \cup \{y^{(k+1)}\}$;
(9) $k \leftarrow k + 1$;
(10) **end**
(11) **end**

---

O algoritmo proposto começa por determinar uma ou duas soluções não-dominadas, consoante o tipo de representação pretendida, se pretendemos uma representação uniforme $\gamma$ ou uma representação $\delta$, correspondentes aos mínimos lexicográficos dos objectivos considerados. De seguida aplica-se o algoritmo de restrição-$\varepsilon$, para

o cálculo da solução não-dominada mais próxima do local pretendido, até que as condições de paragem definidas pelo utilizador sejam verificadas (ver Algoritmo 9).

Nas duas subsecções seguintes apresentamos o algoritmo adaptado a duas formas diferentes de escolher os parâmetros que medem a qualidade da representação. Na primeira secção, admitimos que o utilizador com o apoio do analista pretende um conjunto de soluções não-dominadas correspondentes a uma representação $\gamma$, com $\gamma$ fixo, com um valor $\delta$ tão grande quanto possível. Na segunda secção admitimos que queremos uma representação uniforme $\delta$, $\delta$ fixo, com um valor de $\gamma$ tão pequeno quanto possível.

### 8.2.2 Uma representação $\gamma$

Suponhamos que pretendemos determinar uma representação $\gamma$ do problema FRIBO com um valor de $\delta$ tão grande quanto possível. Neste caso, podemos utilizar o algoritmo de restrição-$\varepsilon$ da seguinte forma:

1. Calcular os mínimos lexicográficos para o problema FRIBO, isto é, as soluções $y^{(1)}$ e $y^*$ tais que

$$y^{(1)} = lex \min_{x_{ij} \in X} \begin{pmatrix} \sum\limits_{(i,j)\in\mathcal{A}} c_{ij}^1 x_{ij} \\ \sum\limits_{(i,j)\in\mathcal{A}} c_{ij}^2 x_{ij} \end{pmatrix} \tag{8.1}$$

   e

$$y^* = lex \min_{x_{ij} \in X} \begin{pmatrix} \sum\limits_{(i,j)\in\mathcal{A}} c_{ij}^2 x_{ij} \\ \sum\limits_{(i,j)\in\mathcal{A}} c_{ij}^1 x_{ij} \end{pmatrix}. \tag{8.2}$$

   Sabemos que ambas as soluções $y^{(1)}$ e $y^*$ são não-dominadas e que têm componentes inteiras. São, por isso, soluções não-dominadas suportadas para o problema FRIBO. Designemos por $Y_r$ a representação do conjunto das soluções não-dominadas inteiras $ND(Y^I)$.

2. Verificar se $y^{(1)}$ e $y^*$ preenchem as condições para pertencer a $Y_r$ e, em caso afirmativo, incluí-las neste conjunto. Verificar se a condição de paragem do algoritmo se verifica, isto é, se todas as soluções não-dominadas estão a uma distância, de uma solução na representação, inferior ou igual a $\gamma$. Caso contrário continuar no passo 3.

3. Fazer $k = 1$.

4. Determinar a solução não-dominada seguinte, $y^{(k+1)}$, a ser incluída na representação $Y_r$. De acordo com a definição escolhida para a qualidade da nossa representação podem ser obtidas diversas soluções. Neste caso, considerar o par de soluções $y^{(k_1)}$ e $y^{(k_2)}$ com maior distância entre si e determinar a solução não-dominada $y^{(k)}$, que está, aproximadamente, à mesma distância de $y^{(k_1)}$ e $y^{(k_2)}$.

5. Repetir o passo anterior, com $k = k + 1$, até que a distância entre qualquer par de soluções não-dominadas seja inferior ou igual a $\gamma$.

Sabemos que uma solução, $y^{(k+1)}$, que satisfaça exactamente as condições acima raramente existe. Procuraremos uma solução próxima, isto é, que diste aproximadamente o mesmo das duas primeiras, resolvendo o problema de restrição-$\varepsilon$ com $\varepsilon = \frac{y_2^{(k_1)} + y_2^{(k_2)}}{2}$, para determinar a solução não-dominada com o maior valor para o segundo objectivo e menor ou igual que $\varepsilon$.

### 8.2.3 Uma representação uniforme $\delta$

Numa representação uniforme $\delta$ as soluções da representação devem distar entre si pelo menos $\delta$ unidades. Além disso, vamos obrigar a que o $\gamma$ seja um valor tão pequeno quanto possível. O algoritmo de restrição-$\varepsilon$ pode ser utilizado da seguinte forma:

1. Calcular o mínimo lexicográfico, $y^{(1)}$, para o problema FRIBO, tal que

$$y^{(1)} = lex \min_{x_{ij} \in X} \begin{pmatrix} \sum\limits_{(i,j)\in\mathcal{A}} c_{ij}^1 x_{ij} \\ \sum\limits_{(i,j)\in\mathcal{A}} c_{ij}^2 x_{ij} \end{pmatrix}, \tag{8.3}$$

   $y^{(1)}$ é uma solução não-dominada para o FRIBO e, inicialmente, $Y_r = \{y^{(1)}\}$.

2. Calcular o mínimo do segundo objectivo, isto é, o valor

$$y_2^* = \min_{x_{ij} \in X} f_2(x)$$

3. Fazer $k = 1$.

4. Determinar a solução não-dominada seguinte, $y^{(k+1)}$, a ser incluída na representação $Y_r$, resolvendo o problema de restrição-$\varepsilon$. Neste caso, a restrição $\varepsilon$ deve obrigar a que a distância da solução $y^{(k+1)}$ a $y^{(k)}$ seja pelo menos igual a

$\delta$. Por exemplo, utilizando a distância euclidiana (ver Figura 8.1), o valor de $\varepsilon$ passa a ser variável, por forma a que $y^{(k+1)}$ esteja o mais próximo possível de $y^{(k)}$, distando pelo menos $\delta$ unidades desta. Assim, para cada número real $y_1 \geqslant y_1^{(k)}$, definimos $\varepsilon(y_1)$ como:

$$\varepsilon(y_1) = \begin{cases} y_2^{(k)} - \sqrt{\delta^2 - (y_1 - y_1^{(k)})^2} & \text{se } y_1^{(k)} \leqslant y_1 < y_1^{(k)} + \delta \\ y_2^{(k)} & \text{se } y_1 \geqslant y_1^{(k)} + \delta \end{cases} .$$

O primeiro conjunto de valores, para $\varepsilon$, corresponde ao quarto de circunferência, representado na Figura 8.1.


**Figura 8.1:** *Representação esquemática de $\varepsilon$.*

5. $(\cdots)$ Repetir o passo anterior, com $k = k + 1$, até não existir qualquer solução não-dominada a uma distância de pelo menos $\delta$ unidades da solução actual, na região não explorada.

O algoritmo de restrição-$\varepsilon$ pode ser facilmente adaptado para trabalhar com este tipo de valores para $\varepsilon$.

## 8.3 Exemplo ilustrativo

Consideremos o problema de fluxos em redes com dois objectivos representado na Figura 8.2. A seguir ilustramos a determinação das representações $\gamma$ e uniforme $\delta$.

### 8.3.1 Uma representação $\gamma$

Suponhamos que, como na Subsecção 8.2.2, queremos uma representação $\gamma$ do conjunto das soluções não-dominadas. Fixemos, ao acaso, $\gamma = 10$.

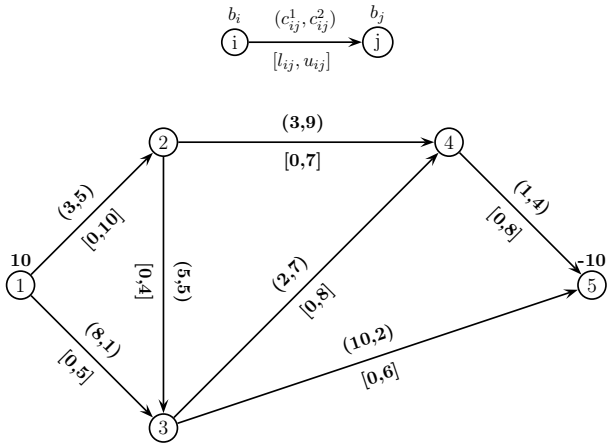



**Figura 8.2:** *Exemplo de uma rede bi-objectivo.*

Apliquemos o algoritmo a este problema:

It.1 O algoritmo começa por calcular as duas soluções óptimas léxicográficas $y^{(1)} = (96, 144)$ e $y^* = (136, 99)$ (estão representadas na Figura 8.3 pelos dois pontos mais espessos).

$y^{(1)}$ e $y^*$ são soluções não-dominadas. A distância entre estas soluções não-dominadas é $d(y^{(1)}, y^*) = 60.2$ e a condição de paragem, considerando uma distância inferior a 10, não é verificada. Assim $Y_r = \{y^{(1)}, y^*\}$. O conjunto $Y_r$ é considerado com os seus elementos ordenados por ordem crescente da primeira coordenada dos seus vectores.

Todas as soluções em $ND(Y^I)$ estão no rectângulo, com os vértices superior esquerdo e inferior direito nos pontos $y^{(1)}$ e $y^*$, respectivamente (ver Figura 8.3).

Consideremos o valor $\varepsilon = \frac{y_2^{(1)}+y_2^*}{2} = 121.5$. De seguida calculamos $y^{(2)}$ tal que

$$\begin{array}{rll} y^{(2)} \leftarrow & \text{minimizar} & f_1(x), \\ & \text{sujeito a:} & x \in X^I, \\ & & f_2(x) \leqslant 121.5. \end{array}$$

A solução óptima para este problema é $x^{(2)} = (6, 4, 0, 6, 0, 4, 6)$ com $y^{(2)} = f(x^{(2)}) = (114, 120)$. O conjunto $Y_r$ é actualizado para $Y_r = \{y^*, y^{(1)}, y^{(2)}\}$. Passamos à iteração seguinte.

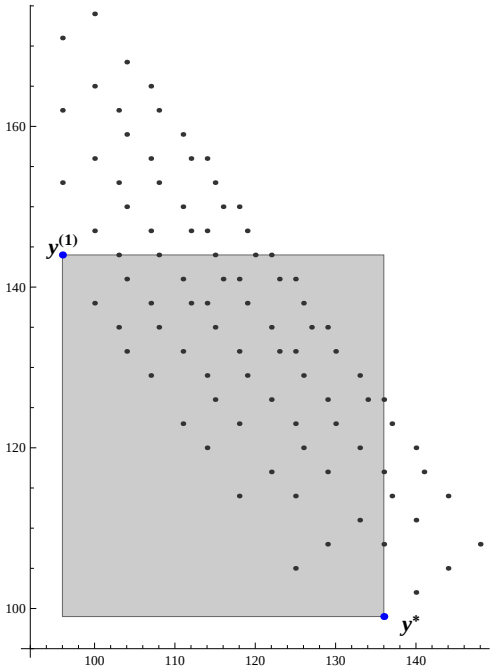



**Figura 8.3:** *Área de procura de soluções não-dominadas.*

It.2 Sabemos agora que o conjunto $ND(Y^I)$ está incluído nos dois rectângulos da Figura 8.4. $(y^{(2)}, y^*)$ é o par de soluções, entre as soluções vizinhas já encontradas, com maior distância (30.4 unidades). Assim é este o par considerado. Coloquemos $\varepsilon = \frac{y_2^{(2)}+y_2^{(*)}}{2} = 109.5$ e resolvamos o problema seguinte

$$\begin{array}{rll} y^{(3)} \leftarrow & \text{minimizar} & f_1(x), \\ & \text{sujeito a:} & x \in X^I, \\ & & f_2(x) \leqslant 109.5. \end{array}$$

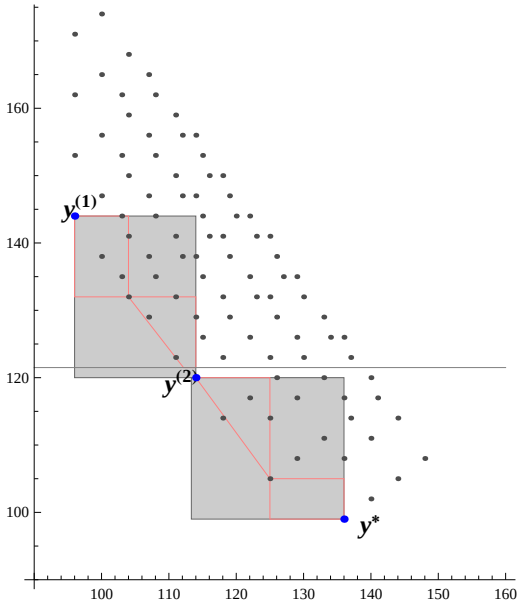



**Figura 8.4:** *Esquema do resultado obtido na 2ª iteração.*

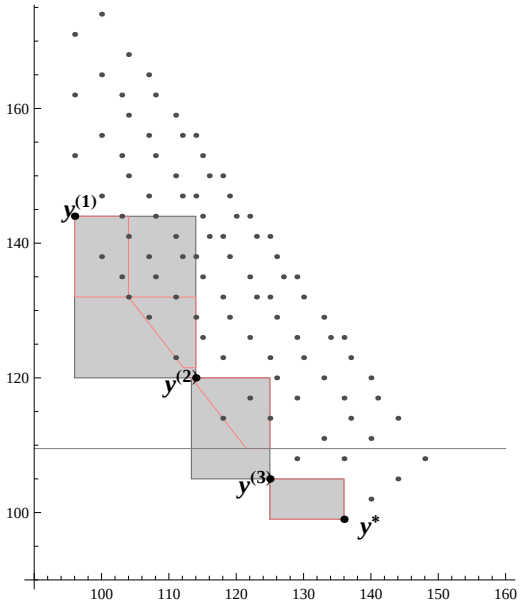



**Figura 8.5:** *Esquema do resultado obtido na 3ª iteração.*

A solução óptima é $x^{(3)} = (5, 5, 0, 5, 0, 5, 5)$ com $y^{(3)} = f(x^{(3)}) = (125, 105)$. O conjunto $Y_r$ é actualizado para $Y_r = \{y^{(1)}, y^{(2)}, y^{(3)}, y^*\}$. A nossa partição tem agora quatro soluções não-dominadas e não temos a garantia de que a distância entre qualquer solução não-dominada e uma solução de $Y_r$ seja inferior a 10. Assim, o algoritmo continua.

It.3 $(\cdots)$ O par de elementos com maior distância é $(y^{(1)}, y^{(2)})$ e $\varepsilon = 132.75$. O problema de restrição-$\varepsilon$ é resolvido e o algoritmo continua até termos a garantia de que para qualquer solução não-dominada em $ND(Y^I)$ existe um elemento em $Y_r$ que dista 10 unidades ou menos daquela.

**Nota:** A forma como o algoritmo foi implementado obriga ao cálculo de outras soluções não-dominadas antes de obter a solução desejada. Essa informação é guardada, para futuros cálculos, reduzindo dessa forma a zona de procura de soluções não-dominadas a um conjunto de trapézios, representados nas Figuras 8.4 e 8.5 pelas linhas a vermelho.

### 8.3.2 Exemplo de uma representação uniforme $\delta$

Determinemos agora uma representação uniforme $\delta$, de acordo com a subsecção 8.2.3. A determinação de $y^{(1)}$ e do mínimo para o segundo objectivo foi feita na subsecção anterior e temos $y^{(1)} = (96, 144)$ e $y_2^* = 99$. Fixemos $\delta = 10$ e $k = 1$.

It.1 Na primeira iteração o algoritmo resolve o problema de restrição-$\varepsilon$ com

$$\varepsilon = \begin{cases} 144 - \sqrt{10^2 - (y_1 - 96)^2} & \text{se } 96 \leqslant y_1 < 106 \\ 144 & \text{se } y_1 \geqslant 106 \end{cases} .$$

A solução obtida é $y^{(2)} = (103, 135)$. A representação actual é então o conjunto das duas soluções não-dominadas encontradas, isto é, $Y_r = \{y^{(1)}, y^{(2)}\}$. Fazemos $k = 2$ e vamos para a iteração seguinte.

It.2 Nesta iteração o valor de $\epsilon$ é:

$$\varepsilon = \begin{cases} 135 - \sqrt{10^2 - (y_1 - 103)^2} & \text{se } 103 \leqslant y_1 < 113 \\ 135 & \text{se } y_1 \geqslant 113 \end{cases} .$$

A resolução do problema de restrição-$\varepsilon$ associado dá origem à solução não-dominada $y^{(3)} = (111, 123)$. Adicionamos esta solução ao conjunto $Y_r$, fazemos $k = 3$ e passamos à iteração seguinte.

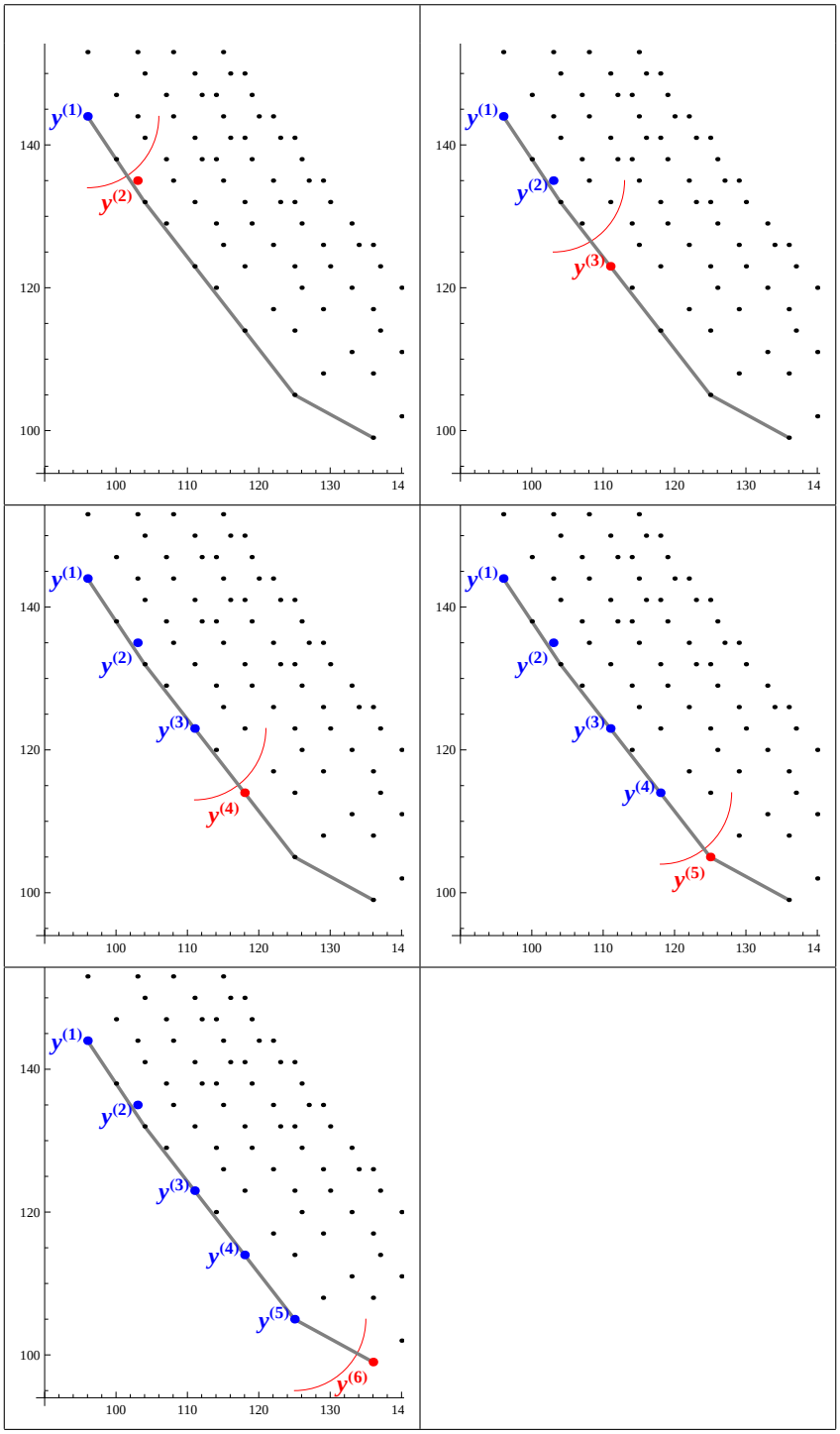



**Figura 8.6:** *Determinação de uma representação uniforme $\gamma$.*

It.3 $(\cdots)$ O algoritmo continua. A representação obtida é o conjunto $Y_r = \{(96, 144), (103, 135), (111, 123), (118, 114), (125, 105), (136, 99)\}$, com seis soluções não--dominadas. A Figura 8.6 mostra as várias etapas da aplicação do algoritmo a este problema.

Em cada iteração é marcado o círculo indicado com um quarto de circunferência e procuramos a solução não-dominada no exterior mas o mais perto possível desse círculo abaixo da recta $y_2 = y_2^{(k)}$. Na primeira iteração, com $y^{(k)} = y^{(1)}$ determinámos $y^{(2)}$. Na segunda iteração, com $k = 2$ determinámos $y^{(3)}$, na terceira, com $k = 3$ determinámos $y^{(4)}$, na quarta, com $k = 4$ determinámos $y^{(5)}$ e, finalmente, na quinta iteração, com $k = 5$, determinámos $y^{(6)}$. A solução não-dominada $y^{(6)}$ é tal que $y_2^{(6)}$ é o mínimo do segundo objectivo quando as restrições do problema inicial são mantidas.

## 8.4 Resultados computacionais

O algoritmo foi implementado usando linguagem de programação C. Os testes foram feitos num computador pessoal equipado com um processador Intel Pentium 2.5GHz com 3GB de RAM. O programa foi executado utilizando o sistema operativo OS X.

O conjunto de problemas e instâncias utilizados nos testes foram os mesmos do Capítulo 7 cujos parâmetros se encontram especificados na Figura 7.12. Na Tabela 8.1 encontram-se os resultados obtidos, para uma representação $\gamma$, como na Subsecção 8.2.2. O valor de $\gamma$, para cada problema e para cada instância, foi calculado automaticamente sendo aproximadamente igual a 10% da distância entre as soluções não-dominadas associadas aos mínimos lexicográficos para cada uma das funções. A escolha foi ao acaso. Olhando para o tempo gasto no cálculo da representação podemos dizer que ele é bastante reduzido, para o conjunto de problemas testados. Em média o algoritmo determinou 9 soluções não-dominadas para cada instância com um número mínimo de 2 e um máximo de 12. Em média o tempo gasto para este cálculo foi de 0.46 segundos, com um tempo mínimo de 0 segundos e um máximo de 8.9 segundos.

**Tabela 8.1:** *Resultados computacionais.*

| | Soluções ND | | | | Tempo *CPU* (seg.) | | | |
|---|---|---|---|---|---|---|---|---|
| N | $\bar{x}$ | Min | Max | Me | $\bar{x}$ | Min | Max | Me |
| 1 | 8.5 | 5 | 12 | 8.5 | 0.01 | 0 | 0.03 | 0 |
| 2 | 8.83 | 5 | 12 | 9 | 0.01 | 0 | 0.02 | 0.01 |
| 3 | 8.9 | 5 | 12 | 9 | 0.01 | 0 | 0.03 | 0.01 |
| 4 | 8.93 | 2 | 12 | 9 | 0.01 | 0 | 0.03 | 0.01 |
| 5 | 8.3 | 3 | 11 | 9 | 0.01 | 0 | 0.02 | 0.01 |
| 6 | 8.07 | 2 | 12 | 8 | 0.01 | 0 | 0.03 | 0 |
| 7 | 9.3 | 6 | 12 | 9 | 0.01 | 0 | 0.03 | 0.01 |
| 8 | 8.83 | 6 | 10 | 9 | 0.01 | 0 | 0.05 | 0.01 |
| 9 | 8.47 | 5 | 12 | 8.5 | 0.04 | 0.01 | 0.1 | 0.04 |
| 10 | 9.47 | 6 | 11 | 10 | 0.05 | 0.01 | 0.1 | 0.04 |
| 11 | 9.6 | 7 | 11 | 10 | 0.06 | 0.01 | 0.14 | 0.05 |
| 12 | 8.6 | 6 | 11 | 9 | 0.07 | 0.01 | 0.49 | 0.05 |
| 13 | 8.93 | 6 | 11 | 9 | 0.04 | 0.01 | 0.18 | 0.04 |
| 14 | 9.2 | 6 | 11 | 9 | 0.05 | 0.01 | 0.12 | 0.05 |
| 15 | 9.37 | 5 | 12 | 10 | 0.06 | 0.02 | 0.2 | 0.05 |
| 16 | 9.43 | 6 | 12 | 9 | 0.07 | 0.02 | 0.21 | 0.06 |
| 17 | 8.93 | 6 | 12 | 9 | 0.1 | 0.04 | 0.23 | 0.09 |
| 18 | 8.7 | 6 | 12 | 8.5 | 0.09 | 0.02 | 0.23 | 0.09 |
| 19 | 9.03 | 6 | 11 | 9 | 0.16 | 0.05 | 0.46 | 0.13 |
| 20 | 9.03 | 6 | 11 | 9 | 0.15 | 0.01 | 0.35 | 0.15 |
| 21 | 9.2 | 7 | 12 | 9 | 0.11 | 0.03 | 0.2 | 0.1 |
| 22 | 8.93 | 6 | 12 | 9 | 0.09 | 0.03 | 0.16 | 0.09 |
| 23 | 9.4 | 7 | 12 | 9 | 0.15 | 0.05 | 0.42 | 0.13 |
| 24 | 9.07 | 6 | 12 | 9 | 0.17 | 0.04 | 0.51 | 0.15 |
| 25 | 9.2 | 6 | 11 | 9.5 | 0.25 | 0.05 | 0.54 | 0.24 |
| 26 | 9 | 6 | 11 | 9.5 | 0.21 | 0.07 | 0.4 | 0.2 |
| 27 | 9.43 | 8 | 11 | 9 | 0.35 | 0.1 | 0.68 | 0.32 |
| 28 | 9 | 7 | 11 | 9 | 0.36 | 0.12 | 0.87 | 0.32 |
| 29 | 9 | 7 | 10 | 9 | 0.23 | 0.04 | 0.45 | 0.21 |
| 30 | 8.93 | 6 | 11 | 9 | 0.2 | 0.07 | 0.53 | 0.16 |
| 31 | 9.23 | 8 | 11 | 9 | 0.38 | 0.15 | 1.35 | 0.3 |
| 32 | 9 | 7 | 11 | 9 | 0.3 | 0.1 | 0.56 | 0.28 |
| 33 | 8.8 | 5 | 11 | 9 | 0.01 | 0 | 0.07 | 0.01 |
| 34 | 8 | 5 | 11 | 8 | 0.01 | 0 | 0.04 | 0.01 |
| 35 | 9 | 5 | 11 | 9 | 0.02 | 0 | 0.1 | 0.01 |
| 36 | 9 | 5 | 12 | 9 | 0.02 | 0.01 | 0.04 | 0.02 |
| 37 | 8.53 | 4 | 12 | 8 | 0.01 | 0 | 0.03 | 0.01 |

*continua na próxima página*

*continuação da página anterior*

| | Soluções ND | | | | Tempo *CPU* (seg.) | | | |
|---|---|---|---|---|---|---|---|---|
| N | $\bar{x}$ | Min | Max | Me | $\bar{x}$ | Min | Max | Me |
| 38 | 8.93 | 7 | 11 | 9 | 0.01 | 0 | 0.05 | 0.01 |
| 39 | 9.07 | 5 | 12 | 9 | 0.02 | 0 | 0.05 | 0.02 |
| 40 | 8.7 | 3 | 11 | 9 | 0.01 | 0 | 0.05 | 0.01 |
| 41 | 7.9 | 6 | 11 | 8 | 0.04 | 0.01 | 0.11 | 0.03 |
| 42 | 8.77 | 5 | 10 | 9 | 0.09 | 0.01 | 0.22 | 0.08 |
| 43 | 9.07 | 6 | 11 | 9 | 0.15 | 0.02 | 0.41 | 0.13 |
| 44 | 9.07 | 6 | 11 | 9 | 0.15 | 0.04 | 0.53 | 0.12 |
| 45 | 8.83 | 7 | 11 | 9 | 0.09 | 0.03 | 0.24 | 0.09 |
| 46 | 8.67 | 6 | 11 | 9 | 0.09 | 0.01 | 0.18 | 0.08 |
| 47 | 9.27 | 7 | 11 | 10 | 0.15 | 0.04 | 0.38 | 0.13 |
| 48 | 9.37 | 7 | 11 | 9.5 | 0.13 | 0.04 | 0.37 | 0.13 |
| 49 | 9.2 | 6 | 11 | 9 | 0.38 | 0.09 | 0.82 | 0.34 |
| 50 | 9.33 | 7 | 11 | 10 | 0.34 | 0.11 | 0.85 | 0.28 |
| 51 | 9.57 | 8 | 12 | 10 | 0.69 | 0.22 | 1.52 | 0.64 |
| 52 | 9.47 | 7 | 12 | 10 | 0.56 | 0.21 | 1.7 | 0.45 |
| 53 | 9.2 | 6 | 11 | 9 | 0.34 | 0.09 | 0.68 | 0.32 |
| 54 | 9.37 | 7 | 11 | 9.5 | 0.32 | 0.04 | 0.76 | 0.27 |
| 55 | 9.3 | 8 | 11 | 9 | 0.55 | 0.19 | 1.37 | 0.53 |
| 56 | 9.13 | 7 | 11 | 9 | 0.52 | 0.13 | 1.51 | 0.45 |
| 57 | 9.23 | 8 | 10 | 9 | 0.72 | 0.31 | 1.53 | 0.7 |
| 58 | 9.17 | 7 | 11 | 9 | 0.65 | 0.26 | 1.17 | 0.68 |
| 59 | 9.7 | 8 | 11 | 10 | 1.75 | 0.35 | 4.33 | 1.52 |
| 60 | 9.47 | 8 | 11 | 9 | 1.07 | 0.14 | 2.63 | 0.92 |
| 61 | 9.13 | 8 | 11 | 9 | 0.74 | 0.31 | 1.47 | 0.74 |
| 62 | 9.07 | 7 | 11 | 9 | 0.7 | 0.27 | 2.29 | 0.65 |
| 63 | 9.4 | 7 | 11 | 9.5 | 1.5 | 0.53 | 3.22 | 1.46 |
| 64 | 9.27 | 7 | 11 | 9 | 1.06 | 0.43 | 1.89 | 1.05 |
| 65 | 9.4 | 7 | 11 | 10 | 1.5 | 0.71 | 2.42 | 1.35 |
| 66 | 9.37 | 7 | 11 | 9 | 1.45 | 0.4 | 2.74 | 1.37 |
| 67 | 9.47 | 8 | 11 | 9 | 3.32 | 1.17 | 8.94 | 3.32 |
| 68 | 9.27 | 7 | 11 | 9.5 | 2.02 | 0.97 | 6.1 | 1.66 |
| 69 | 8.9 | 7 | 10 | 9 | 1.48 | 0.42 | 4.51 | 1.48 |
| 70 | 9.37 | 7 | 12 | 9.5 | 1.56 | 0.62 | 3.55 | 1.43 |
| 71 | 9.13 | 7 | 11 | 9 | 3.12 | 1.37 | 7.37 | 3.13 |
| 72 | 9.4 | 7 | 11 | 9 | 2.34 | 0.94 | 3.9 | 2.27 |

# Conclusão e investigação futura

Os problemas multi-objectivo de fluxos em redes ocupam uma área muito importante no âmbito do problema tratado e da Investigação Operacional, pela sua ligação a um enorme número de problemas reais, de que apresentámos exemplos na Introdução desta Tese, e também pelo facto de aparecerem como subproblemas de outros problemas, justificando-se, deste modo, a investigação levada a cabo por nós.

Em resumo, nesta Tese estudámos os problemas de fluxos em redes com mais de um objectivo, propusemos novas formas de calcular o conjunto das soluções eficientes/não-dominadas e novas formas de determinar aproximações para este conjunto. Apresentámos dois novos algoritmos do tipo primal-dual, um para o problema de fluxo de custo mínimo e outro para o problema bi-objectivo de fluxos em redes, ambos com variáveis contínuas. O primeiro algoritmo apareceu como consequência dos estudos feitos sobre o segundo e os resultados obtidos mostram que, na resolução de problemas gerados aleatoriamente, o tempo gasto pode ser inferior ao tempo gasto pelo algoritmo simplex para redes. O segundo determina o conjunto de todas as soluções não-dominadas extremas e a sua implementação revelou bons resultados, sobretudo, na resolução de problemas altamente degenerados. As restantes soluções não-dominadas podem ser obtidas como combinações convexas de pares de soluções não-dominadas extremas.

Relativamente ao problema inteiro multi-objectivo de fluxos em redes estudámos o conjunto das soluções suportadas e provámos que este é um conjunto conexo. Esta prova foi feita através da utilização de redes residuais e de circuitos de custo nulo nestas redes. Com base na demonstração elaborámos um algoritmo que permite determinar o conjunto de todas as soluções eficientes/não-dominadas suportadas para este problema. O algoritmo foi implementado e experimentado para um número elevado de problemas e instâncias e os resultados foram apresentados e revelaram um bom comportamento do algoritmo em termos computacionais. A utilização de circuitos de custo nulo para a determinação de soluções não-dominadas foi, tanto quanto é do nosso conhecimento, feito aqui pela primeira vez. Mostrámos também, através de exemplos e ilustrando com figuras, que o conjunto de todas as soluções não-dominadas suportadas nem sempre é formado apenas pela imagem de soluções ex-

tremas e soluções intermédias, contrariamente ao que era assumido até aqui. Ainda, para estes problemas, apresentámos uma nova versão do algoritmo de restrição-$\varepsilon$, para o cálculo de todas as soluções eficientes e não-dominadas, mostrando que a sua aplicação pode ser feita a qualquer problema, mesmo a problemas degenerados. Fizemos a sua implementação e executámos o código para uma quantidade elevada de problemas e instâncias, fazendo depois o estudo estatístico dos dados e resultados. Finalmente, ainda para estes problemas propusemos um algoritmo que determina aproximações para o conjunto de todas as soluções não-dominadas. Este algoritmo pode gerar vários tipos de representações do conjunto das soluções não-dominadas, em função da definição de "boa" representação. Fizemos a sua implementação e apresentámos os resultados. É sabido que tanto o conjunto de soluções eficientes, como o conjunto de soluções não-dominadas têm uma cardinalidade, em geral, muito elevada, de tal forma que qualquer análise desse conjunto se torna um trabalho quase impossível, assumindo, por isso, um grande relevo o cálculo de "boas" representações desse conjunto. No entanto, a existência de algoritmos exactos, que consigam calcular o conjunto de todas as soluções, é uma ajuda importante na construção de algoritmos para o cálculo de uma "boa" representação.

O trabalho apresentado deixa em aberto um conjunto de pistas de investigação para o futuro, algumas das quais descrevemos a seguir. Em primeiro lugar, parece que muito do trabalho aqui efectuado pode ser generalizado para problemas com mais de dois objectivos. Assim, tanto o algoritmo primal-dual como o algoritmo de restrição-$\varepsilon$ podem ser estendidos a problemas com três objectivos, numa primeira fase, e provavelmente ser generalizados para o caso de um número $p > 3$ de objectivos. Pensamos que o algoritmo primal-dual poderá ser vantajoso, relativamente aos paramétricos, não só para casos degenerados, mas mesmo no caso geral. Este trabalho será de grande importância, uma vez que não se conhecem algoritmos especialmente dedicados a problemas com mais de dois objectivos. O algoritmo de circuito de custo zero, como foi apresentado, calcula apenas as soluções suportadas para o problema multi-objectivo de fluxos em redes. Parece-nos, no entanto, que é possível transformar este algoritmo de forma a que possa calcular o conjunto de todas as soluções eficientes ou não-dominadas. Esta tarefa será igualmente de grande importância, uma vez que se trata de um algoritmo multi-objectivo, que revelou um bom comportamento computacional. Todo este trabalho poderá ser integrado e ser programado num sistema de apoio ao utilizador, para resolução de problemas de fluxos em redes. Este sistema ajudará o utilizador na escolha de uma "boa" representação do conjunto de soluções eficientes ou não-dominadas para um problema deste tipo. Tal sistema deverá conseguir mostrar ao utilizador características do conjunto a representar, como estimativas da cardinalidade desse conjunto, da distância média das soluções da representação às soluções não calculadas, etc.

Na conclusão desta Tese podemos afirmar que os objectivos a que nos propusemos foram plenamente cumpridos superando até as nossas expectativas iniciais, na medida em que foram detectadas e resolvidas algumas falhas nesta área e é proposto um conjunto de ferramentas que acreditamos vir a servir de base a novos resultados.

# Referências


Ahuja, R., Magnanti, T.L., & Orlin, J. 1993. *Network Flows: Theory, Algorithms, and Applications*. New Jersey: Prentice Hall.

Aneja, Y.P., & Nair, K.P.K. 1979. Bicriteria transportation problem. *Management Science*, **25**(1), 73–78.

Bazaraa, M.S., Jarvis, J.J., & Sherali, H.D. 2005. *Linear Programming and Network Flows*. 3rd edn. New Jersey: John Wiley & Sons, New York.

Benson, H.P., & Sun, E. 2000. Outcome space partition of the weight set in multiobjective linear programming. *Journal of Optimization Theory & Applications*, **105**(1), 17–36.

Burkard, R.E., Hamacher, H.W., & Rote, G. 1987. *Approximation of convex functions and applications in mathematical programming*. Germany. Report 89-1987, TU Graz, Institut für Mathematik.

Burkard, R.E., Hamacher, H.W., & Rote, G. 1991. Sandwich approximation of univariate convex functions with an application to separable convex programming. *Naval Research Logistics Quarterly*, **38**(6), 911–924.

Chankong, V., & Haimes, Y.Y. 1983. *Multiobjective Decision Maker: Theory and Methodology*. New York: North-Holland.

Coffin, M., & Saltzman, M.J. 2000. Statistical analysis of computational tests of algorithms and heuristics. *INFORMS Journal on Computing*, **12**(1), 24–44.

Cunningham, W.H. 1976. A network simplex method. *Mathematical Programming*, **11**, 105–106.

DaCunha, N.O., & Polak, E. 1967. Constrained minimization under vector-valued criteria in finite dimensional spaces. *Journal of Mathematical Analysis & Applications*, **19**, 103–124.

Dauer, J.P., & Gallagher, R.J. 1996. A combined constraint-space, objective space approach for determining high-dimensional maximal efficient faces of multiple objective linear programs. *European Journal of Operational Research*, **88**, 368–381.

Ehrgott, M. 1999. Integer solutions of multicriteria network flow problems. *Investigação Operacional*, **19**, 229–243.

Ehrgott, M. 2005. *Multicriteria Optimization*. 2nd edn. Berlin: Springer-Verlag.

Ehrgott, M., Puerto, J., & Rodríguez-Chía, A.M. 2007. A primal-dual simplex method for multiobjective linear programming. *Journal of Optimization Theory and Applications*, **134**, 483–497.

Eusébio, A., & Figueira, J.R. 2006. *Bi-Criteria network flow problems: A characterization of non-dominated solutions. Centro de Estudos de Gestão do Instituto Superior Técnico*, Working Paper nº3/2006.

Eusébio, A., & Figueira, J.R. 2007. *A note on the computation of nondominated supported vectors in bicriteria network flow problems. Centro de Estudos de Gestão do Instituto Superior Técnico*, Working Paper nº3/2007.

Eusébio, A., & Figueira, J.R. 2008. *A negative-cycle algorithm for computing all the supported efficient solutions in multi-objective integer network flow problems. Centro de Estudos de Gestão do Instituto Superior Técnico*, Working Paper nº11/2008.

Eusébio, A., & Figueira, J.R. 2009a. Finding non-dominated solutions in bi-objective integer network flow problems. *Computers & Operations Research*, **36**, 2554–2564.

Eusébio, A., & Figueira, J.R. 2009b. On the computation of all supported efficient solutions in multi-objective integer network flow problems. *European Journal of Operational Research*, **199**, 68–76.

Eusébio, A., Figueira, J.R., & Ehrgott, M. 2008. *A primal-dual simplex algorithm for bi-objective network flow problems. Quarterly Journal of Operations Research*, doi: 10.1007/s10288-008-0087-3.

Figueira, J.R. 2000. *On the integer bi-criteria network flows problem: A branch-and--bound approach*. Faculdade de Economia da Universidade de Coimbra, Coimbra, Portugal. Working Paper No. 3/2000 [revised version: Cahier du LAMSADE, Nº 191/2002, LAMSADE, Université Paris-Dauphine, Paris, France].

Figueira, J.R. 2001. *Filtering algorithm for bi-criteria network flows.* Manuscript.

Fruhwirth, B., Burkard, R.E., & Rote, G. 1989. Approximation of convex curves with application to the bicriterial minimum cost flow problem. *European Journal of Operational Research*, **42**(3), 326–338.

Gal, T. 1977. A general method for determining the set of all efficient solutions to a linear vectormaximum problem. *European Journal of Operational Research*, **1**, 307–322.

Geoffrion, A.M. 1967. Solving bicriterion mathematical programs. *Operations Research*, **15**(1), 39–54.

Geoffrion, A.M. 1968. Proper efficiency and the theory of vector maximization. *Journal of Math Analysis and Applications*, **22**, 618–630.

Gouveia, M.C. 2002. *Bi-criteria network flow problems: The study of an algorithm for searching efficient solutions [in Portuguese].* M.Phil. thesis, Faculdade de Economia da Universidade de Coimbra, Coimbra, Portugal.

Grigoriadis, M.D. 1986. An efficient implementation of the network simplex method. *Mathematical Programming Study*, **26**, 83–111.

Haimes, Y., Lasdon, L., & Wismer, D. 1971. On a bicriterion formulation of the problems of integrated system identification and system optimization. *IEEE Transactions on Systems, Man, and Cybernetics*, **1**, 296–297.

Hamacher, H.W. 1995. A note on $K$ best network flows. *Annals of Operations Research*, **57**, 65–72.

Hamacher, H.W., Pedersen, C.R., & Ruzika, S. 2007a. Finding representative systems for discrete bicriterion optimizations problems. *Operations Research Letters*, **35**(3), 336–344.

Hamacher, H.W., Pedersen, C.R., & Ruzika, S. 2007b. Multiple objective minimum cost flow problems: A review. *European Journal of Operational Research*, **176**, 1404–1422.

Huarng, F., Pulat, P., & Ravindran, A. 1992. An algorithm for bicriteria integer flow problem. *Pages 305 – 318 of: 10th International Conference on Multiple Criteria Decision Making*, vol. 3.

Isermann, H. 1977. The enumeration of the set of all efficient solutions for a linear multiple objective program. *Operational Research Quarterly*, **28**(3), 711–725.

Klingman, D., Napier, A., & Stutz, J. 1974. NETGEN: A program for generating large scale capacitated assignment, transportation, and minimum cost flow network problems. *Management Science*, **20**(5), 814–821.

Lee, H., & Pulat, P.S. 1991. Bicriteria network flow problems: Continuous case. *European Journal of Operational Research*, **51**, 119–126.

Lee, H., & Pulat, P.S. 1993. Bicriteria network flow problems: Integer case. *European Journal of Operational Research*, **66**, 148–157.

Lim, G.J., & Lee, E.K. 2008. *Optimization on Medecine and Biology*. New York: Auerbach Publications.

Malhotra, R., & Puri, M.C. 1984. Bi-criteria network problem. *Cahiers du C.E.R.O.*, **26**(1), 95–102.

Mustafa, A., & Goh, M. 1998. Finding integer efficient solutions for bicriteria and tricriteria network flow problems using DINAS. *Computers & Operations Research*, **25**(2), 139–157.

Naccache, P. 1978. Connectedness of the set of nondominated outcomes in multicriteria optimization. *Journal of Optimization Theory and Applications*, **25**, 459–467.

Nikolova, M. 1998. Properties of the effective solutions of the multicriteria network flow problem. *Problems of Engineering Cybernetics and Robotics*, **47**, 104–111.

Nikolova, M. 2001. An approach for finding Pareto optimal solutions of the multicriteria network flow problem. *Cybernetics and Information Technologies*, **2**, 56–62.

Nikolova, M. 2003. A classification based approach for finding Pareto optimal solutions of the multicriteria network flow. *Cybernetics and Information Technologies*, **3**(1), 11–16.

Ogryczak, W., Studzinski, K., & Zorychta, K. 1992. DINAS: A computer-assisted analysis system for multiobjective transhipment problems with facility location. *Computers and Operations Research*, **19**(7), 637–648.

Papadimitriou, C.H. 1982. *Combinatorial Optimization: Algorithms and Complexity*. New Jersey: Prentice Hall.

Przybylski, A., Gandibleux, X., & Ehrgott, M. 2006. The biobjective integer minimum cost flow problem - incorrectness of Sedeño-Noda and González-Martín's algorithm. *Computers & Operations Research*, **33**(5), 1459–1463.

Pulat, P.S., Huarng, F., & Lee, H. 1992. Efficient solutions for the bicriteria network flow problem. *Computers & Operations Research*, **19**(7), 649–655.

Raith, A., & Ehrgott, M. 2009. A two-phase algorithm for the biobjective integer-minimum cost flow problem. *Computers & Operations Research*, **36**, 1945–1954.

Ruhe, G. 1988a. Complexity results for multicriterial and parametric network flows using a pathological graph of Zadeh. *Zeitschrift für Operations Research*, **32**, 9–27.

Ruhe, G. 1988b. *Flüsse in Netzwerken - Komplexität und Algorithmen*. Ph.D. thesis, Technische Hochschule Leipzig, Sektion Mathematik und Informatik, Germany.

Ruhe, G., & Fruhwirth, B. 1990. $\epsilon$-optimality for bicriteria programs and its application to minimum cost flows. *Computing*, **44**, 21–34.

Ruzika, S., & Wiecek, M.M. 2005. Approximation methods in multiobjective programming. *Journal of Optimization Theory and Applications*, **126**(3), 473–501.

Sayin, S. 1999. Measuring the quality of discrete representations of efficient sets in multiple objective mathematical programming. *Mathematical Programming*, **87**, 543–560.

Schechter, M., & Steuer, R.E. 2005. A correction to the connectedness of the Evans-Steuer algorithm of multiple objective linear programming. *Foundations of Computing and Decision Sciences*, **30**(4), 351–359.

Schrijver, A. 2000. *Theory of Linear and Integer Programming*. New York: John Wiley & Sons.

Sedeño Noda, A., & González-Martín, C. 2000. The biobjective minimum cost flow problem. *European Journal of Operational Research*, **124**, 591–600.

Sedeño Noda, A., & González-Martín, C. 2001. An algorithm for the biobjective integer minimum cost flow problem. *Computers & Operations Research*, **28**, 139–156.

Sedeño Noda, A., & González-Martín, C. 2003. An alternative method to solve the biobjective minimum cost flow problem. *Asia-Pacific Journal of Operational Research*, **20**, 241–260.

SPSS. 2007. *SPSS 16.0 Brief Guide*. Chicago: SPSS Inc.

Steuer, R.E. 1986. *Multiple Criteria Optimization: Theory, Computation, and Application*. New York: John Wiley & Sons.

Warburton, A.R. 1983. Quasiconcave vector maximization: Connectedness of the sets of Pareto-optimal and weak Pareto-optimal alternatives. *Journal of Mathematical Analysis & Applications*, **40**(4), 537–557.

Yang, X.Q., & Goh, C.J. 1997. A method for convex curve approximation. *European Journal of Operational Research*, **97**, 205 – 212.

Zeleny, M. 1974. *Linear Multiobjective Programming. Lecture Notes in Economics and Mathematical Systems*. Vol. 95. New York: Springer-Verlag.