Instituto Federal de Educação, Ciência e Tecnologia da Paraíba

Campus João Pessoa

Programa de Pós-Graduação em Tecnologia da Informação

Nível Mestrado Profissional

KERVEN MACIEL MONTEIRO DE ALBUQUERQUE

MÉTODOS EXATOS E HEURÍSTICOS PARA O PROBLEMA DO FLUXO COM ROTULAÇÃO MÍNIMA

DISSERTAÇÃO DE MESTRADO

JOÃO PESSOA

2024

Kerven Maciel Monteiro de Albuquerque

Métodos exatos e heurísticos para o problema do fluxo com rotulação mínima

Dissertação apresentada como requisito parcial para obtenção do título de Mestre em Tecnologia da Informação, pelo Programa de Pós-Graduação em Tecnologia da Informação do Instituto Federal da Paraíba – IFPB.

Orientador: Prof. Dr. Thiago Gouveia

João Pessoa

2024

Dados Internacionais de Catalogação na Publicação – CIP
Biblioteca Nilo Peçanha – IFPB, *campus* João Pessoa

A345m Albuquerque, Kerven Maciel Monteiro de.

Métodos exatos e heurísticos para o problema do fluxo com rotulação mínima / Kerven Maciel Monteiro de Albuquerque. – 2023.

53 f. : il.

Dissertação (Mestrado em Tecnologia da Informação) – Instituto Federal da Paraíba – IFPB / Programa de Pós-Graduação em Tecnologia da Informação - PPGTI.

Orientador: Prof. Dr. Thiago Gouveia.

1. Grafos com Arestas Rotuladas (GAR). 2. Corte coloridos. 3. Fluxo máximo. 4. Métodos exatos. 5. Métodos heurísticos. 6. Experimentos computacionais. I. Título.

CDU 004.02

Bibliotecária responsável Taize Araújo da Silva – CRB15/536

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA

PROGRAMA DE PÓS-GRADUAÇÃO *STRICTO SENSU*

MESTRADO PROFISSIONAL EM TECNOLOGIA DA INFORMAÇÃO

KERVEN MACIEL MONTEIRO DE ALBUQUERQUE

MÉTODOS EXATOS E HEURÍSTICOS PARA O PROBLEMA DO FLUXO COM ROTULAÇÃO MÍNIMA

Dissertação apresentada como requisito para obtenção do título de Mestre em Tecnologia da Informação, pelo Programa de Pós- Graduação em Tecnologia da Informação do Instituto Federal de Educação, Ciência e Tecnologia da Paraíba – IFPB - Campus João Pessoa.

Aprovado em 16 de janeiro de 2024

Membros da Banca Examinadora:

Dr. Thiago Gouveia da Silva

IFPB - PPGTI

Dr. Ruan Delgado Gomes

IFPB - PPGTI

Dr. Gilberto Farias de Sousa Filho

UFPB

João Pessoa/2024

RESUMO

Um grafo com arestas rotuladas (GAR) é um grafo no qual cada aresta é associada a um rótulo, ou cor. Esses rótulos permitem representar características qualitativas, em contraste às características quantitativas que podem ser expressas por custos ou capacidades. A literatura tem dado atenção crescente a problemas definidos sobre GARs. Este trabalho introduz o problema do fluxo com rotulação mínima (PFRM), definido sobre um dígrafo com arcos rotulados e capacitados. Nele, busca-se um conjunto mínimo de rótulos tal que seja possível encontrar um fluxo com valor igual ou superior a um valor dado. São propostos métodos exatos e heurísticos para o PFRM. Como métodos exatos, são propostos uma formulação baseada em cortes coloridos e dois algoritmos de branch and cut. Como métodos heurísticos, um GRASP reativo e um algoritmo genético melhorado com construção gulosa aleatorizada baseada em GRASP, elitização e recombinação por caminhos, além de um método de manutenção dinâmica de fluxo. Os experimentos computacionais realizados demonstraram a eficiência dos métodos propostos em instâncias de tamanho moderado, em comparação com as soluções encontradas na literatura.

Palavras-chaves: Grafos com arestas rotuladas. Fluxo máximo. Cortes coloridos.

ABSTRACT

An edge-labeled graph (ELG) is a graph in which each edge has a label, or color, associated with it. Such labels allow representing qualitative characteristics, in contrast to the quantitative characteristics represented by costs or capacities. In the literature, there has been increasing attention to problems defined over ELGs. This work presents the minimum labeling flow problem (MLFP), defined over a directed capacitated and labeled graph. It aims for a minimum set of labels such that a flow with a value equal to or greater than a given value exists. Heuristic and exact methods for the MLFP are proposed. Regarding exact methods, a formulation based on colorful cuts and two branch-and-cut algorithms are proposed. As heuristic methods, a Reactive GRASP and an improved genetic algorithm, with a GRASP-based greedy randomized construction, elitist selection, and crossover based on path-relinking, along with a dynamic flow maintenance method. Computational experiments have shown the efficiency of the proposed methods in medium-sized instances, compared to the solutions found in the literature.

Keywords: Edge-labeled graphs. Maximum flow. Colorful cuts.

LISTA DE FIGURAS

Figura 1 – Exemplos de DARC (a) e respectivas soluções para o PFM (b), para o PFMRM (c), e para o PFRM com valor desejado de fluxo igual a 6 (d). . . . 13
Figura 2 – Exemplo de aplicação do PFRM em uma rede (a) e solução otimizando a infraestrutura para utilizar apenas 2 rótulos, garantindo um fluxo de valor 5 (b). 15
Figura 3 – Exemplo de aplicação do PFMRM no projeto de uma CDN onde os arcos que partem da origem representam as demandas dos clientes (a), e solução suprindo todas as demandas com apenas 3 rótulos (c). 15
Figura 4 – Exemplo de instância do PAGRM (a), redução para uma instância do PFMRM (b), e solução do PFMRM que corresponde a uma solução equivalente do PAGRM (c). 23

LISTA DE TABELAS

Tabela 1 – Resultados dos experimentos com os métodos exatos em grafos de 100 vértices. 38
Tabela 2 – Resultados dos experimentos com os métodos exatos em grafos de 200 vértices. 39
Tabela 3 – Resultados dos experimentos com os métodos heurísticos em grafos de 50 vértices. 41
Tabela 4 – Resultados dos experimentos com os métodos heurísticos em grafos de 100 vértices. 41
Tabela 5 – Resultados dos experimentos com os métodos heurísticos em grafos de 200 vértices. 42
Tabela 6 – Resultados dos métodos de fluxo (em ms), agrupados por $|L|$, com $f_g = f_{max}$. 44
Tabela 7 – Resultados dos métodos de fluxo (em ms), agrupados pelo fluxo dado, com $|L| = |V|$. 45
Tabela 8 – Resultados dos métodos de fluxo (em ms), por densidade, sem arcos paralelos. 46
Tabela 9 – Resultados dos métodos de fluxo (em ms), agrupados por arcos paralelos, com $d = 0{,}8$. 47

LISTA DE ALGORITMOS

1 Algoritmo de Edmonds-Karp 21
2 Algoritmo Push-Relabel com seleção da maior altura 22

3 Algoritmo de *branch and cut* para a FCC 29
4 Algoritmo de *branch and cut* para a FCC em duas etapas 30

5 Etapa de construção gulosa aleatorizada 31
6 Etapa de busca local 32
7 GRASP Reativo 32
8 Algoritmo genético melhorado 34
9 Cálculo do fluxo máximo a partir de um fluxo dado 35
10 Cálculo do fluxo máximo após a remoção de um rótulo 35

LISTA DE ABREVIATURAS E SIGLAS

AGM	Algoritmo genético melhorado
CDN	*Content delivery network*, ou rede de distribuição de conteúdo
DAC	Dígrafo com arcos capacitados
DARC	Dígrafo com arcos rotulados e capacitados
FCC	Formulação baseada em cortes coloridos
FF	Formulação baseada em fluxo
GAR	Grafo com arestas rotuladas
PAAIS	Problema da árvore arco-íris de Steiner
PAGRM	Problema da árvore geradora de rotulação mínima
PCMC	Problema do corte mínimo colorido
PFAIP	Problema da floresta arco-íris panorâmica
PFM	Problema do fluxo mínima
PFMRM	Problema do fluxo máximo com rotulação mínima
PFRM	Problema do fluxo com rotulação mínima
PLI	Programção linear inteira
PLIM	Programação linear inteira mista
PMCC	Problema do menor caminho colorido
PMCkR	Problema do menor caminho com k rótulos
RC	Rede de computadores

SUMÁRIO

1 INTRODUÇÃO 12
1.1 Definição do problema 13
1.2 Motivação 14
1.3 Objetivos 15
1.3.1 Objetivo geral 15
1.3.2 Objetivos específicos 16
1.4 Estrutura do documento 16

2 FUNDAMENTAÇÃO TEÓRICA 17
2.1 Grafos 17
2.1.1 Fluxo máximo e corte mínimo 17
2.1.2 Algoritmos de fluxo 20
2.1.3 Dígrafos com arcos rotulados 22
2.2 Trabalhos relacionados 24
2.2.1 Formulação baseada em fluxo 26

3 MÉTODOS EXATOS 28
3.1 Formulação baseada em cortes coloridos 28
3.2 Algoritmos de *branch and cut* 29

4 MÉTODOS HEURÍSTICOS 31
4.1 GRASP reativo 31
4.2 Algoritmo genético melhorado 33
4.3 Manutenção dinâmica do fluxo 34

5 EXPERIMENTOS COMPUTACIONAIS 37
5.1 Métodos exatos 37
5.2 Métodos heurísticos 40
5.3 Manutenção dinâmica de fluxo 42
5.4 Discussão 43

6 CONSIDERAÇÕES FINAIS 49

REFERÊNCIAS BIBLIOGRÁFICAS 51

1 INTRODUÇÃO

Um grafo com arestas rotuladas (GAR) é um grafo no qual cada aresta é associada a um rótulo, ou cor. Esses rótulos permitem representar características qualitativas, em contraste às características quantitativas que podem ser expressas por custos ou capacidades. Problemas definidos sobre GARs têm ganhado atenção crescente na literatura, como pode-se observar nos trabalhos de Granata et al. (2013) e Silva (2018).

Um dos problemas de maior destaque na área é o problema da árvore geradora de rotulação mínima (PAGRM). Ele consiste em, dado um GAR, determinar um subconjunto mínimo de rótulos que permita obter uma árvore geradora (SILVA et al., 2015; SILVA, 2018). Outros exemplos incluem o problema do diâmetro colorido (FIGUEIRÊDO et al., 2019; FIGUEIRÊDO et al., 2020) e o problema do corte global rotulado mínimo (SILVA et al., 2018).

Este trabalho aborda o problema do fluxo com rotulação mínima (PFRM). Esse problema é definido sobre um grafo orientado, ou dígrafo, com arcos rotulados e capacitados (DARC). Dados um DARC, um valor desejado de fluxo e vértices de origem e destino, o PFRM busca obter um subconjunto mínimo de rótulos tal que, no subgrafo induzido, exista um fluxo com valor igual ao desejado. Caso o valor desejado seja do fluxo máximo, trata-se do problema do fluxo máximo com rotulação mínima (PFMRM), abordado por Granata et al. (2013).

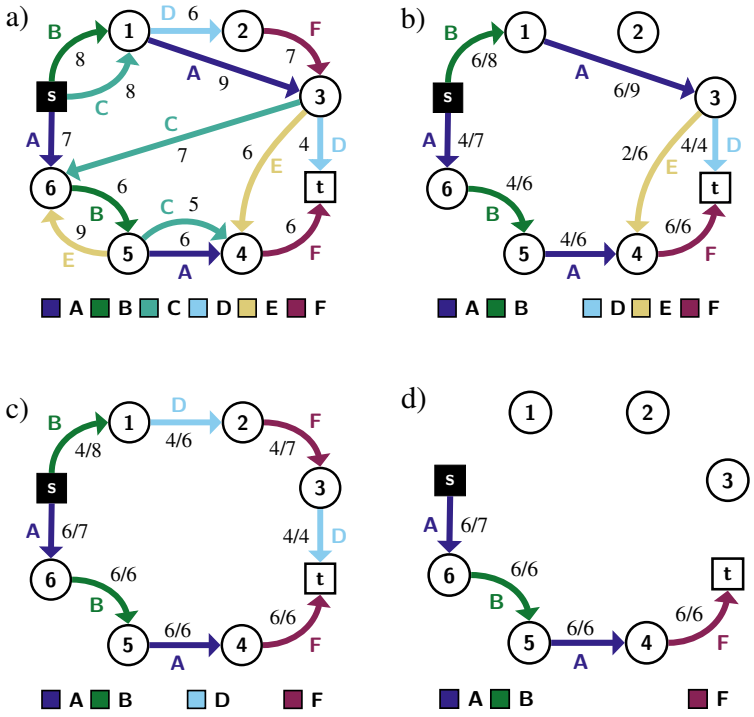
A Figura 1 exemplifica soluções desses problemas. Um DARC com 8 vértices e 14 arcos, é mostrado em (a). A cada arco, estão associados uma capacidade e um dentre 6 rótulos. O valor do fluxo máximo entre os vértices s e t é 10, como ilustrado em (b) e (c). Pode-se obter esse fluxo utilizando 4 rótulos, de forma ótima. Sendo assim, (c) é uma solução do PFMRM em (a). Por sua vez, (d) apresenta uma solução para o PFRM em (a), dado um valor esperado de fluxo igual a 6. Nesse caso, apenas 3 rótulos foram necessários.

Granata et al. (2013) apresentam o PFMRM como uma variação do problema do fluxo máximo (PFM) no contexto de GARs e propõem uma formulação baseada em fluxo (FF) e um método heurístico baseado na meta-heurística SVNS. Eles demonstram que o PFMRM é NP-difícil, por redução do PAGRM. Consequentemente, o PFRM também o é.

Por serem problemas NP-difíceis, métodos exatos conhecidos não são eficientes e tornam-se rapidamente impraticáveis com o crescimento do grafos. Nesse caso, as formulações e melhorias propostas para os métodos exatos observam o tempo médio de execução, ainda que a complexidade no pior caso permaneça exponencial. Outra abordagem é abrir mão da exatidão em troca do desempenho de métodos heurísticos.

Este trabalho propõe-se a introduzir o PFRM e apresentar as primeiras soluções. Como solução exata, propomos uma adaptação para o PFRM da formulação baseada em cortes coloridos (FCC) de Silva (2018), melhorada através de algoritmos de *branch and cut*, como apresentado

Figura 1 – Exemplos de DARC (a) e respectivas soluções para o PFM (b), para o PFMRM (c), e para o PFRM com valor desejado de fluxo igual a 6 (d).



Fonte: autoria própria.

em Albuquerque et al. (2021). A FCC foi originalmente proposta para o PAGRM e adaptada para vários problemas, incluindo o PFMRM. Como soluções heurísticas, são propostos um GRASP reativo e um algoritmo genético melhorado (AGM), com construção gulosa aleatorizada baseada na meta-heurística GRASP, elitização e recombinação por caminhos, apresentado em Albuquerque, Oliveira e Silva (2020).

Entre os resultados teóricos, será apresentada uma prova de que o PFRM é NP-difícil, além de uma análise da relação entre o PFMRM e o PFRM. Também será analisado um método de manutenção dinâmica de fluxo, proposto para melhorar o desempenho dos algoritmos heurísticos, adaptando o fluxo máximo a adições e remoções de rótulos, em vez de recalculá-lo a partir de um fluxo vazio, como apresentado em Albuquerque, Oliveira e Silva (2020).

1.1 Definição do problema

Seja um grafo orientado e capacitado $D = (V, A)$, no qual cada arco $a \in A$ está associado a uma capacidade $c_a \geq 0$. Sejam, ainda, vértices de origem $s \in V$ e destino $t \in V$. Um fluxo s-t em D (ou simplesmente um fluxo) é uma função $f : A \to \mathbb{R}$ que associa a cada arco $a \in A$ um

valor real f_a satisfazendo (AHUJA et al., 2013):

$$\sum_{a \in \delta_v^+} f_a - \sum_{a \in \delta_v^-} f_a = \begin{cases} |f|, & \text{para } v = s \\ 0, & \text{para todo } v \in V \setminus \{s,t\} \\ -|f|, & \text{para } v = t \end{cases} \quad (1)$$

$$0 \leq f_a \leq c_a, \text{ para todo } a \in A \quad (2)$$

Os conjuntos $\delta_v^+ = \{a = (v,u) \mid a \in A, u \in V\}$ e $\delta_v^- = \{a = (u,v) \mid a \in A, u \in V\}$ denotam respectivamente os arcos que saem do vértice v e aqueles que terminam em v. O número $|f| \geq 0$ é chamado de valor do fluxo. A equação (1) é a lei de conservação do fluxo, que garante que o fluxo total em cada vértice é nulo, à exceção da origem e destino. Em outras palavras, o fluxo que entra num vértice deve ser igual ao que sai dele.

Antes de definir o PFRM, convém introduzir mais uma notação. Dado um dígrafo com arcos rotulados (DAR) $D = (V,A,L)$ e um subconjunto de rótulos $L' \subseteq L$, $D[L']$ representa o subgrafo gerador de D induzido pelo conjunto de arcos $A[L'] = \{a \in A \mid l_a \in L'\}$. Pode-se, então, enunciar o problema.

Definição 1. *Seja um DARC $D = (V,A,L)$, no qual cada arco $a \in A$ está associado a uma capacidade $c_a \geq 0$ e um rótulo $\ell_a \in L$. Sejam, ainda, vértices $s,t \in V$ e um valor $f_g \geq 0$ esperado de fluxo s-t. O PFRM objetiva encontrar um subconjunto mínimo de rótulos $L' \subseteq L$ tal que exista um fluxo s-t f em $D[L']$ com valor $|f| \geq f_g$.*

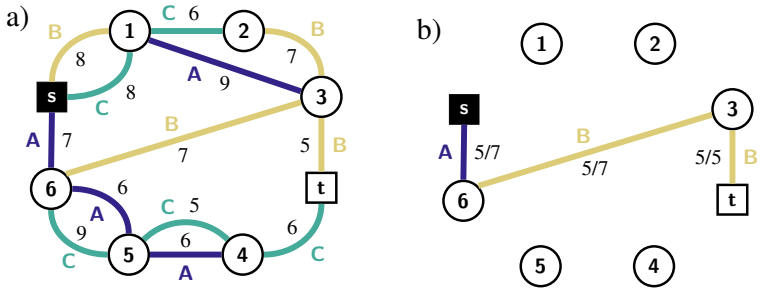
1.2 Motivação

O PFRM tem potencial de aplicação em diversas áreas do conhecimento, teóricas e práticas. Um exemplo no contexto das Redes de Computadores (RC) consiste na definição da infraestrutura mínima necessária para suprir a demanda prevista por uma rede. Considere que uma empresa contrata sua infraestrutura de rede de outras empresas. É interessante que a quantidade de empresas contratadas seja minimizada a fim de reduzir os custos de operação, mas sem prejuízo da qualidade do serviço. A Figura 2 exemplifica essa aplicação.

Na figura, o grafo à esquerda modela todas as possibilidades de uma rede. A demanda a ser suprida é o valor dado de fluxo entre os nós s e t da rede. As arestas podem ser agrupadas de acordo com seus rótulos, que representam as redes que poderão ser contratadas. A solução do PFRM, neste caso, é apresentada no grafo à direita. Pela quantidade de cores utilizadas, seria necessário contratar duas empresas para suprir a demanda.

Uma aplicação semelhante envolve a seleção ótima dos servidores para uma rede de distribuição de conteúdo (CDN, do inglês, *content delivery network*). De acordo com as características de cada servidor, delimita-se o grupo de clientes que podem ser satisfatoriamente

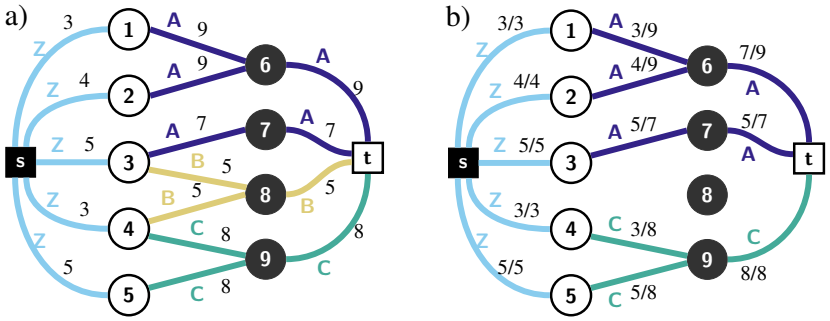
Figura 2 – Exemplo de aplicação do PFRM em uma rede (a) e solução otimizando a infraestrutura para utilizar apenas 2 rótulos, garantindo um fluxo de valor 5 (b).



Fonte: autoria própria.

supridos. A partir da demanda de tráfego, o PFRM determina a menor quantidade de provedores necessários, diminuindo custos. Isso está exemplificado na Figura 3.

Figura 3 – Exemplo de aplicação do PFMRM no projeto de uma CDN onde os arcos que partem da origem representam as demandas dos clientes (a), e solução suprindo todas as demandas com apenas 3 rótulos (c).



Fonte: autoria própria.

No grafo à esquerda, os nós brancos representam grupos de clientes, enquanto os nós cinzas representam os servidores de distribuição de conteúdo. Tais servidores estão agrupados de acordo com o provedor, representado pelo rótulo nas arestas. Cada cliente possui uma demanda, indicada pelas capacidades das arestas que saem da origem s. Cada servidor é capaz de suprir a demanda indicada na aresta que o liga ao destino t.

A solução do PFRM com fluxo dado $f_g = f_{max} = 20$ é representada no grafo à direita. Os vértices s e t não representam entidades da CDN, mas são um recurso teórico para definir o sentido do fluxo. No exemplo, três rótulos são necessários para garantir f_g. O rótulo Z serve apenas para conectar os clientes à origem do fluxo. Sendo assim, dois provedores é o mínimo necessário para suprir a demanda.

1.3 Objetivos

1.3.1 Objetivo geral

Introduzir o PFRM e propor soluções exatas e heurísticas eficientes.

1.3.2 Objetivos específicos

- Analisar o PFRM, incluindo a relação com o PFMRM e uma prova de que é NP-difícil.
- Propor uma FCC para o PFRM, baseada na solução de Silva (2018) para o PFMRM, e melhorada com algoritmos de *branch and cut*.
- Propor um GRASP reativo e um AGM, com construção gulosa aleatorizada baseada em GRASP, elitização e recombinação por caminhos.
- Comparar os resultados dos métodos propostos com a FF e o SVNS de Granata et al. (2013) para o PFMRM.
- Propor e analisar um método de manutenção dinâmica de fluxo para melhorar o desempenho dos algoritmos heurísticos.

1.4 Estrutura do documento

Este documento está dividido em 6 capítulos, incluindo esta introdução. Os demais capítulos estão organizados como segue.

- O Capítulo 2 apresenta a fundamentação teórica necessária à compreensão do trabalho, explicando conceitos envolvendo grafos e o PFRM, além de analisar os trabalhos relacionados.
- O Capítulo 3 descreve a FCC e os algoritmos de *branch and cut* propostos.
- O Capítulo 4 descreve os métodos heurísticos propostos, a saber, o GRASP reativo e o AGM, além do método de manutenção dinâmica de fluxo.
- O Capítulo 5 discute os experimentos computacionais realizados.
- O Capítulo 6 apresenta as considerações finais do trabalho e propostas de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, são discutidos alguns fundamentos teóricos que fornecem a base para o desenvolvimento do trabalho. A Seção 2.1 define conceitos em grafos, incluindo fluxo máximo e seus principais algoritmos, corte mínimo, grafos com arcos rotulados, além de resultados teóricos diretamente relacionados ao PFRM. Então, a Seção 2.2 apresenta os principais trabalhos relacionados encontrados na literatura, com destaque para os métodos heurísticos e exatos propostos para o PFMRM.

2.1 Grafos

Seja um grafo orientado, ou dígrafo, $D = (V,A)$. Um arco $a \in A$, que parte de um vértice $u \in V$ para um vértice $v \in V$, pode ser representado pelo par ordenado (u,v). Valores associados aos arcos podem ser representados por funções, mesmo que sejam omitidas por simplicidade. Um dígrafo com arcos capacitados (DAC), por exemplo, considera uma função de capacidades $c : a \in A \mapsto c_a \geq 0$, que associa capacidades não-negativas aos arcos. Um DAR $D = (V,A,L)$, por sua vez, considera uma rotulação $\ell : a \in A \mapsto \ell_a \in L$.

Esta seção descreve os conceitos de fluxo e corte e alguns resultados teóricos importantes, como o teorema do fluxo máximo e corte mínimo. Também são apresentadas as principais estratégias para o cálculo do fluxo máximo e os algoritmos utilizados neste trabalho. Conclui-se com conceitos em DARs e resultados básicos para o problema.

2.1.1 Fluxo máximo e corte mínimo

Sejam um grafo orientado $D = (V,A)$ e um vértice $v \in V$. O conjunto dos arcos que saem de v é representado por $\delta_v^+ = \{a = (v,u) \mid a \in A, u \in V\}$, enquanto o conjunto dos arcos que chegam a v é representado por $\delta_v^- = \{a = (u,v) \mid a \in A, u \in V\}$. Para um conjunto $S \subseteq V$ de vértices, tem-se que $\delta_S^+ = \{(u,v) \in A \mid u \in S, v \notin S\}$ e $\delta_S^- = \{(u,v) \in A \mid u \notin S, v \in S\}$.

No caso de funções reais definidas sobre os arcos do grafo, uma notação semelhante será utilizada para representar o somatório da função nos arcos de entrada ou saída de um vértice. Dada uma função $x : a \in A \mapsto x_a \in \mathbb{R}$ e um vértice $v \in V$, tem-se $x_v^+ = \sum_{a \in \delta_v^+} x_a$ e $x_v^- = \sum_{a \in \delta_v^-} x_a$. Analogamente, dado $S \subseteq V$, tem-se $x_S^+ = \sum_{a \in \delta_S^+} x_a$ e $x_S^- = \sum_{a \in \delta_S^-} x_a$.

Essa notação é baseada em Bondy e Murty (1976) e representa o total de uma função na entrada ou saída de um vértice. Em um DAC, por exemplo, c_v^- representa a capacidade total de entrada do vértice v, ou seja, a soma das capacidades de todos os arcos que entram em v. Analogamente, dado $S \subseteq V$, c_S^+ é a capacidade total de saída do conjunto S. Então, pode-se enunciar a definição de fluxo.

Definição 2. *Sejam um DAC $D = (V,A)$ e dois vértices $s,t \in V$ distintos. Um **fluxo** s-t em D (ou simplesmente um fluxo) é uma função $f : a \in A \mapsto f_a \in \mathbb{R}$ tal que*

$$0 \leq f_a \leq c_a, \ \forall a \in A \tag{3}$$

$$f_v^+ - f_v^- = 0, \ \forall v \in V \setminus \{s,t\}. \tag{4}$$

Os vértices s e t são chamados de origem e destino do fluxo, respectivamente. Os demais vértices são intermediários. Em cada vértice intermediário, a lei de conservação do fluxo, apresentada na Equação 4, garante que o fluxo que entra nele é igual ao fluxo que sai.

Fixado um fluxo *s-t* em D, f, faz sentido considerar a capacidade restante dos arcos, além do fluxo f. A capacidade residual de um arco $a = (u,v) \in V$ em relação ao fluxo f é definida por $r_a = c_a - f_a + f_{a'}$, onde $f_{a'}$ é o fluxo no arco reverso $a' = (v,u)$, se existir. Dado um caminho P em D, diremos que sua capacidade residual é a menor dentre seus arcos, ou seja, $r_P = \min_{a \in A_P} r_a$, onde A_P é o conjunto de arcos em P.

Saturar um caminho consiste em aumentar o fluxo no caminho até a capacidade residual tornar-se zero. Para isso, pode-se definir um novo fluxo g tal que $g_{a'} = f_{a'} - r_P$, $g_a = f_a$, caso $f_{a'} \geq r_P$, ou $g_{a'} = 0$, $g_a = f_a + r_P - f_{a'}$, caso $f_{a'} < r_P$. Em ambos os casos, $g_a = f_a$ para quaisquer outros arcos. Assim, remove-se fluxo do arcos reversos do caminho e adiciona-se aos arcos diretos até o limite dado pela capacidade residual do caminho e, consequentemente, esta se tornará zero em relação ao novo fluxo.

Tratando-se da conservação do fluxo, pode-se estendê-la de um único vértice para um conjunto $S \subseteq V \setminus \{s,t\}$ qualquer de vértices intermediários, como apresenta a Proposição 1.

Proposição 1. *Dados um DAC $D = (V,A)$, com um fluxo f entre os vértices s-t em D e um subconjunto $S \subseteq V \setminus \{s,t\}$ de vértices intermediários, tem-se $f_S^+ - f_S^- = 0$.*

Demonstração. No caso de S conter um único vértice v, segue diretamente da definição de fluxo. Então suponha, por indução, que a conservação é mantida para $R = S \setminus \{v\}$, dado $v \in S$ qualquer. Pode-se dizer que $\delta_S^+ = \left(\delta_R^+ \setminus \delta_v^-\right) \cup \left(\delta_v^+ \setminus \delta_R^-\right)$, ou seja, o conjunto de saída de S é composto pelos arcos que saem de R e os que saem de v, excetuando os que estão entre R e v. Segue que $f_S^+ = f_R^+ - f\left(\delta_R^+ \cap \delta_v^-\right) + f_v^+ - f\left(\delta_v^+ \cap \delta_R^-\right)$, onde $f(X) = \sum_{a \in X} f_a$. Analogamente, conclui-se que $f_S^- = f_R^- - f\left(\delta_R^- \cap \delta_v^+\right) + f_v^- - f\left(\delta_v^- \cap \delta_R^+\right)$. Subtraindo as equações, obtém-se $f_S^+ - f_S^- = f_R^+ - f_R^- + f_v^+ - f_v^-$. Portanto, $f_S^+ - f_S^- = 0$. □

Poder-se-ia acrescentar à Definição 2 que o fluxo resultante que sai da origem, $f_s^+ - f_s^-$, deve ser igual ao fluxo resultante que chega ao destino, $f_t^- - f_t^+$, como em Ahuja et al. (2013). Isso também pode ser visto como uma consequência da Proposição 1. Tomando $X = V \setminus \{s,t\}$, tem-se $f_X^+ - f_X^- = 0$. Além disso, $\delta_X^+ = \delta_s^- \cup \delta_t^-$ e $\delta_s^- \cap \delta_t^- = \emptyset$, logo $f_X^+ = f_s^- + f_t^-$. De forma análoga, conclui-se que $f_X^- = f_s^+ + f_t^+$. Portanto, $f_s^- - f_s^+ = f_t^+ - f_t^-$. Essa quantidade é

chamada de valor do fluxo e representada por $|f|$. É de especial interesse quando esse valor é máximo, conforme enunciado na Definição 3.

Definição 3. *Sejam um DAC $D = (V,A)$, vértices $s,t \in V$ e um fluxo s-t f. Diz-se que f é um* ***fluxo máximo*** *quando não existe fluxo s-t f' em D tal que $|f'| > |f|$.*

As definições consideraram fluxos com uma única origem e um único destino. No entanto, pode-se falar de fluxo com múltiplas origens e destinos. Para tal, dado um DAC $D = (V,A)$, um conjunto de origens $S \subseteq V$ e um conjunto de destinos $T \subseteq V \setminus S$, pode-se construir um novo DAC $D' = (V',A')$ tal que $V' = V \cup \{s,t\}$ e $A' = A \cup \delta_s^+ \cup \delta_t^-$, onde $\delta_s^+ = \{(s,v) \mid v \in S\}$, $\delta_t^- = \{(v,t) \mid v \in T\}$ e $c_a = \infty$ para todo $a \in \delta_s^+ \cup \delta_t^-$. Ou seja, adiciona-se um novo vértice de origem s, que é conectado a todas as origens de S com capacidade infinita, assim como um novo vértice de destino t, para o qual todos os destinos de T convergem com capacidade infinita. Um fluxo entre os conjuntos S e T em D, então, equivale a um fluxo s-t em D'.

Um conceito intimamente relacionado ao fluxo é o de corte, apresentado na Definição 4.

Definição 4. *Sejam um DAC $D = (V,A)$ e vértices $s,t \in V$ distintos. Um* ***corte*** *s-t em D (ou simplesmente um corte) é o conjunto de arcos $K = \delta_S^+ \cup \delta_S^-$ definido por uma partição de V em dois subconjuntos (S,T) tal que $s \in S$ e $t \in T$.*

Um corte K será comumente representado pela partição (S,T) que o define. Os arcos no conjunto δ_S^+ são chamados diretos, enquanto os arcos em δ_S^- são chamados reversos. A capacidade de um corte é a soma das capacidades dos seus arcos diretos, ou $c_K = \sum_{a \in \delta_S^+} c_a$. Quando essa capacidade é mínima, tem-se um corte mínimo, definido a seguir.

Definição 5. *Sejam um DAC $D = (V,A)$, vértices $s,t \in V$ e um corte s-t K. Diz-se que K é um* ***corte mínimo*** *quando não existe corte s-t K' em D tal que $c_{K'} < c_K$.*

Por definição, o fluxo em cada arco não excede a sua capacidade. Essa relação se estende para o valor do fluxo e a capacidade de um corte, como descreve a Proposição 2.

Proposição 2. *Sejam um DAC $D = (V,A)$ e vértices $s,t \in V$ distintos. Para qualquer fluxo f e qualquer corte $K = (S,T)$, tem-se $|f| \leq c_K$.*

Demonstração. Por definição, $f_s^+ - f_s^- = |f|$ e $f_v^+ - f_v^- = 0$, para todo $v \in S \setminus \{s\}$. Somando essas equações, obtém-se $f_S^+ - f_S^- = \sum_{v \in S} (f_v^+ - f_v^-) = |f|$. Como $0 \leq f_a \leq c_a$ para cada $a \in K$, então $f_S^+ \leq c_K$ e $f_S^- \geq 0$. Juntando os resultados, $|f| = f_S^+ - f_S^- \leq c_K$. □

A Proposição 2 afirma que a capacidade de qualquer corte é um limite superior para o valor de qualquer fluxo. Em particular, a capacidade do corte mínimo $c_{\min} \in \mathbb{R}^+$ é um limite superior para o valor do fluxo máximo $f_{\max} \in \mathbb{R}^+$. Assim, dados um fluxo f e um corte K tais que $|f| = c_K$, tem-se que f é um fluxo máximo e K é um corte mínimo. Um resultado mais

forte é obtido pelo teorema do fluxo máximo e corte mínimo, por Dantzig e Fulkerson (1955), apresentado a seguir.

Teorema 1 (Fluxo máximo e corte mínimo)**.** *Em qualquer DAC,* $f_{\max} = c_{\min}$.

Demonstração. Seja um DAC $D = (V, A)$ e vértices $s, t \in V$ distintos. Dado um fluxo máximo f, a primeira parte da prova mostra que todos os caminhos *s*-*t* em D estão saturados, ou seja, sua capacidade residual é nula. Suponha, por absurdo, que exista um caminho *s*-*t* P em D com capacidade residual $r_P > 0$ em relação a f. Saturando esse caminho, obtém-se um fluxo f' de valor $|f'| = |f| + r_P$, o que contradiz f ser um fluxo máximo.

Seja, então, um fluxo f tal que todos os caminhos *s*-*t* em D estejam saturados. Pode-se definir o conjunto S dos vértices v tais que existe um caminho *s*-*v* não saturado. Tem-se $s \in S$ e $t \notin S$, logo podemos considerar o corte $K = (S, T = V \setminus S)$. Qualquer arco direto de K está saturado. De fato, se existisse $a = (u, v) \in \delta_K^+$ tal que $f_a < c_a$, então poderíamos tomar um caminho *s*-*u* não saturado e estendê-lo com a até v, que implicaria em $v \in S$, o que é absurdo. Logo, $f_S^+ = c_K$. O mesmo raciocínio pode ser empregado para mostrar qualquer arco reverso de K tem fluxo nulo, logo, $f_S^- = 0$. Usando essas igualdades na prova da Proposição 2, conclui-se que $|f| = c_K$. Portanto, f é um fluxo máximo e K um corte mínimo. Como qualquer fluxo máximo satisfaz a definição de f, conclui-se que $f_{\max} = c_{\min}$. □

O Teorema 1 é fundamental na Teoria dos Grafos e muitos resultados derivam dele. Um desses resultados é a FCC discutida neste trabalho. A prova apresentada, baseada em Bondy e Murty (1976), contém o protótipo de um método construtivo para encontrar um fluxo máximo. Ele consiste em partir de um fluxo anterior, encontrar um caminho não saturado e saturá-lo. O processo inicia com um fluxo trivial, nulo, e é repetido até que todos os caminhos estejam saturados, constituindo um fluxo máximo. Esse método foi proposto por Ford e Fulkerson (1956) e uma versão mais refinada é discutida na Seção 2.1.2.

2.1.2 Algoritmos de fluxo

Há duas classes principais de algoritmos para encontrar um fluxo máximo em um DAC (GRANATA et al., 2013). A primeira classe baseia-se em caminhos, buscando caminhos e saturando-os até que não existam caminhos ainda não saturados. A segunda classe utiliza um pré-fluxo, uma função que satisfaz a restrição da capacidade, mas não necessariamente a conservação do fluxo, permitindo excessos nos vértices. Nesse caso, garante-se que ao final do algoritmo, o pré-fluxo terá se tornado num fluxo de fato.

O método proposto por Ford e Fulkerson (1956) é um exemplo da primeira classe. No entanto, ele não diz especifica como encontrar um caminho não saturado. Nesse sentido, Edmonds e Karp (1972) propuseram utilizar uma busca em largura (BFS, do inglês, *breadth-first search*) no grafo residual. O grafo residual é uma abstração na qual considera-se a capacidade residual

dos arcos, em vez da capacidade total. Assim, o grafo residual é formado apenas pelos caminhos não saturados. Esse procedimento é apresentado no Algoritmo 1.

Algoritmo 1: Algoritmo de Edmonds-Karp

```
1 Procedimento EDMONDS-KARP(D = (V,A), s, t)
2     Seja R o grafo residual de D após um fluxo vazio (R = D);
3     Enquanto houver caminho s-t em R faça
4         Seja P o caminho resultante de uma BFS entre s e t em R;
5         Sature P, atualizando o grafo residual R;
6     Retorne o fluxo f referente ao grafo residual resultante R;
```

Na linha 2, o grafo residual é inicializado igual a D. Então, o procedimento consiste em executar uma BFS em R para encontrar um caminho s-t não saturado e saturá-lo, enquanto for possível, conforme linhas 3–5. Por fim, a linha 6 retorna o fluxo correspondente ao grafo residual final, que é máximo, como pôde ser visto na prova do Teorema 1. Edmonds e Karp (1972) mostraram que a complexidade desse algoritmo é $O(|V|\,|A|^2)$.

Em relação à classe de algoritmos que utilizam um pré-fluxo, o principal exemplo é o algoritmo *push-relabel*, proposto por Goldberg e Tarjan (1988). Ele utiliza uma função de alturas $h : v \in V \mapsto h_v \in \mathbb{N}$ para os vértices, de modo que $h_s = |V|$, $h_t = 0$ e, para todo arco $a = (u,v)$ com $r_a > 0$, $h_v \leq h_u + 1$. Note-se que r_a representa a capacidade residual de a em relação a um pré-fluxo f. A partir da função de alturas, definem-se as operações *push* e *relabel*.

Como trata-se de um pré-fluxo, não é garantida a conservação do fluxo em cada vértice e eles podem possuir um excesso de fluxo $e_v = f_v^- - f_v^+$. A operação *push* pode ser aplicada em um arco $a = (u,v)$ e consiste o máximo do excesso de fluxo, ou seja, $\min\{r_a, e_u\}$ através de a, diminuindo e_u e aumentando e_v. Essa operação só pode ser executada quando $e_u > 0$ e $h_u = h_v + 1$, e envia o fluxo do vértice mais alto para um vértice um nível abaixo.

A operação *relabel*, por sua vez, atualiza as alturas de um vértice. Seja um vértice $v \notin \{s,t\}$ com excesso $e_v > 0$. Considere também que $h_v \leq h_u$ para todo $u \in H_v$, onde $H_v = \{u \in V \mid r_a > 0, a = (v,u) \in \delta_v^+\}$. Então, o *relabel* atribui a v a altura $h'_v = \min_{u \in H_v} h_u + 1$. Neste caso, quando um vértice possui excesso, mas sua altura não permite *push* para nenhum dos vizinhos, aumenta-se a altura até o menor valor que permita tal operação.

A versão genérica do *push-relabel* consiste em executar operações de *push* e *relabel* enquanto alguma delas for possível. Inicialmente, o pré-fluxo é definido como $f_a = c_a, \forall a \in \delta_s^+$ e $f_a = 0$ nos demais casos. Já a função de alturas é tal que $h_s = |V|$ e $h_v = 0$ para todo $v \neq s$. Essa versão tem complexidade $O(|V|^2\,|A|)$ e, ao finalizar a execução do algoritmo, o pré-fluxo f é um fluxo máximo (GOLDBERG; TARJAN, 1988).

Pode-se combinar ambas as operações de *push* e *relabel* em uma operação *discharge*. Dado um vértice v, o *discharge* tenta executar *push* em todos os vértices vizinhos, então executa um *relabel* e repete o processo, enquanto v ainda possuir excesso positivo. Pode-se, então,

sempre selecionar o vértice com maior altura e que ainda possui excesso, para executar *discharge*. Isso está representado no Algoritmo 2. Com essas otimizações, obtém-se uma complexidade $O(|V|^2 \sqrt{|A|})$ (CHERIYAN; MAHESHWARI, 1988).

Algoritmo 2: Algoritmo Push-Relabel com seleção da maior altura

```
1 Procedimento PUSH-RELABEL(D = (V,A), s, t)
2     Seja h uma função de alturas, com h_s = |V| e h_v = 0 para qualquer outro vértice
        v;
3     Seja f um pré-fluxo s-t, com f_a = c_a para a ∈ δ_s^+ e f_a = 0 para os demais arcos ;
4     Enquanto houver algum vértice v com excesso e_v > 0 faça
5         Seja um vértice u com e_u > 0 e a maior altura, ou seja, ∄v, e_v > 0, h_v > h_u;
6         DISCHARGE(D, h, f, u);
7     Retorne f ;
```

No algoritmo, as linhas 2–3 inicializam a função de alturas e o pré-fluxo. Então, o laço das linhas 4–6 seleciona um vértice com excesso e altura máxima para executar uma operação de *discharge*, enquanto for possível. Note-se que o *discharge* atualiza as alturas e o pré-fluxo. Ao final do processo, obtém-se um fluxo máximo f, que é retornado na linha 7.

Outra melhoria que pode ser acrescentada ao Algoritmo 2 é inicializar a função de alturas com uma BFS partindo de t. Embora não afete a complexidade, essa melhoria heurística garante que haja um caminho de altura decrescente da origem para o destino, podendo resultar numa diminuição da quantidade de iterações necessárias para os excessos alcançarem o destino.

2.1.3 Dígrafos com arcos rotulados

Dado um conjunto de vértices $V' \subseteq V$, o subgrafo de D induzido pelos vértices em V' será denotado por $D[V']$. Essa notação estende-se para um conjunto de rótulos $L' \subseteq L$, de modo que $D[L']$ representa o subgrafo gerador de D induzido pelo conjunto $A' = \{a \in A \mid \ell_a \in L'\}$ dos arcos com rótulo em L'. A partir dessa notação, alguns conceitos podem ser apresentados.

Definição 6. *Um rótulo $\ell \in L$ é necessário para o fluxo dado f_g (ou simplesmente necessário) se o valor do fluxo máximo f'_{max} no subgrafo $D[L \setminus \{\ell\}]$ for menor que f_g. O conjunto dos rótulos necessários é representado por $L_N \subseteq L$.*

Definição 7. *Um vértice $v \in V$ é necessário para o fluxo dado f_g (ou simplesmente necessário) se o valor do fluxo máximo f'_{max} no subgrafo $D[V \setminus \{v\}]$ for menor que f_g. O conjunto dos vértices necessários é representado por $V_N \subseteq V$.*

As definições de rótulos e vértices necessários são introduzidas por Granata et al. (2013), como elementos sem os quais é impossível obter o fluxo máximo. No entanto, ser necessário para o fluxo máximo não garante a necessidade para um valor menor de fluxo. Por isso, as definições apresentadas são variações, que consideram um fluxo dado. A importância de tais definições está

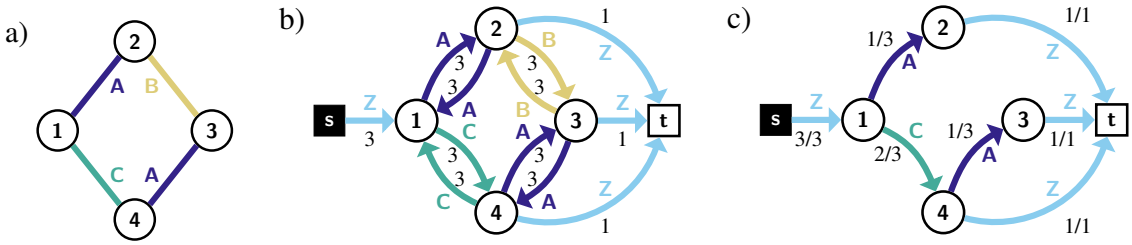
na garantia de que esses elementos estarão presentes na solução ótima, reduzindo o espaço de busca. Granata et al. (2013) baseiam-se nisso para melhorar o desempenho de suas formulações, algo particularmente útil por se tratar de um problema difícil, como mostra a Proposição 3.

Proposição 3. *O PFMRM é NP-difícil.*

Demonstração. Conforme apresentado por Granata et al. (2013), pode-se reduzir o PAGRM ao PFMRM da forma descrita a seguir. Dado um grafo $G = (V, E, L)$, onde cada aresta $e \in E$ está associada a um rótulo $\ell_e \in L$, construa-se um grafo orientado $D' = (V', A', L')$ tal que $V' = V \cup \{s, t\}$ e $L' = L \cup \{\ell'\}$. Então, forme-se o conjunto de arcos A'. Para cada aresta $e = (i, j) \in E$, sejam os arcos $(i, j), (j, i) \in A'$, ambos com capacidade $|V| - 1$ e rótulo ℓ_e. Fixe-se um vértice $u \in V$ qualquer e seja $(s, u) \in A'$ com capacidade $|V| - 1$ e rótulo ℓ'. Por fim, para cada $v \in V \setminus \{u\}$, seja $(v, t) \in A'$ com capacidade 1 e rótulo ℓ'. Note-se que $|V'| = |V| + 2$, $|A'| = |A| + |V|$ e $|L'| = |L| + 1$, logo D' pode ser construído em tempo polinomial.

A Figura 4 exemplifica essa construção. Cada aresta no grafo em (a) corresponde a dois arcos no grafo orientado em (b), de mesmo rótulo e capacidade $|V| - 1 = 3$. Além disso, tomado $u = 1$, há um arco (s, u) de rótulo $\ell' = \mathrm{Z}$ e capacidade 3 e arcos com origem em todos os vértices, exceto u, com destino t, capacidade 1 e rótulo Z. A solução para o PFMRM em (c) tem fluxo de valor 3 e utiliza três rótulos. Observe-se que, ao desconsiderar os vértices s e t e o rótulo Z, permanece a solução para o PAGRM, utilizando dois rótulos.

Figura 4 – Exemplo de instância do PAGRM (a), redução para uma instância do PFMRM (b), e solução do PFMRM que corresponde a uma solução equivalente do PAGRM (c).



Fonte: Adaptado de Granata et al. (2013).

Generalizando, se G é conexo, o fluxo máximo em D' tem valor $|V| - 1$. Tal fluxo inicia-se em s e segue a direção de u aos demais vértices do grafo, cada um contribuindo com uma unidade de fluxo ao destino. Note-se que, considerando apenas os vértices em V, forma-se uma árvore geradora que terá rotulação mínima quando, e somente quando, o fluxo correspondente também utilizar uma quantidade mínima de rótulos. Portanto, uma solução do PFMRM com $k + 1$ rótulos corresponde diretamente a uma solução do PAGRM com k rótulos. □

Como o PFRM é uma generalização do PFMRM, segue da Proposição 3 que o PFRM também é NP-difícil. De fato, a prova pode ser replicada para o PFRM fixando-se $f_{\mathrm{g}} = f_{\max}$. Pode-se ir além na relação entre o PFRM e o PFMRM, como mostra a Proposição 4.

Proposição 4. *O PFRM é redutível ao PFMRM.*

Demonstração. Dada uma instância do PFRM, pode-se construir uma instância do PFMRM substituindo a origem s por um novo vértice s' e adicionando um arco $a' = (s', s)$ com capacidade f_{g} e um novo rótulo ℓ'. O fluxo máximo nessa instância tem valor f_{g}, logo a solução do PFMRM corresponde diretamente a uma solução do PFRM, desconsiderados s', a' e ℓ'. □

Observa-se que a adaptação entre os problemas utilizou-se de apenas um vértice, um arco e um rótulo, podendo ser realizada em tempo constante. Isso permite questionar a relevância de propor soluções para o PFRM em detrimento de utilizar soluções para o PFMRM. No entanto, deve-se considerar que limitar o fluxo máximo no grafo pode diminuir a quantidade de rótulos e vértices necessários. Se $f_{\mathrm{g}} = 0{,}75 f_{\max}$, por exemplo, então apenas os rótulos e vértices pelos quais passa mais de 25% de $f_{\max}$ permanecerão necessários.

A diminuição dos elementos necessários acarreta na perda de otimizações propostas para as formulações. Essa redução de desempenho deverá ser evidenciada nos experimentos computacionais. De todo modo, pode-se dizer que o PFRM corresponde a um caso particularmente lento do PFMRM, o que justifica destacar o seu estudo em busca de propor novos métodos e melhorias, tendo por base a literatura para o PFMRM.

2.2 Trabalhos relacionados

A literatura discute muitos problemas definidos sobre GARs. A maioria deles pertence a uma de três classes: problemas arco-íris, que desejam encontrar conjuntos de arestas com rótulos distintos entre si e satisfazendo uma dada propriedade; problemas de k rótulos, que buscam conjuntos de arestas utilizando até k rótulos distintos; e problemas de rotulação mínima ou máxima, como o PFRM, que buscam um conjunto mínimo (ou máximo) de rótulos que satisfazem a propriedade em questão.

O problema da árvore arco-íris de Steiner (PAAIS) é um exemplo de problema do primeiro tipo. Ele consiste em encontrar uma árvore arco-íris que cobre todos os vértices de um dado conjunto e foi proposto por Ferone, Festa e Guerriero (2022), que modelaram o problema como um multifluxo. Eles propuseram um algoritmo *multi-start* e uma relaxação Lagrangiana como abordagens heurísticas. Também mostraram que o problema é NP-difícil por uma redução simples do problema da árvore de Steiner clássico.

Outro exemplo é o problema da floresta arco-íris panorâmica (PFAIP), que objetiva encontrar uma floresta panorâmica usando o menor número de árvores arco-íris. O primeiro modelo em programação linear inteira (PLI) para o problema foi proposto por Carrabs et al. (2018), assim como um algoritmo guloso. Esse modelo foi melhorado por Moreno, Martins e Frota (2020), que também propuseram uma abordagem heurística com GRASP. Outras soluções heurísticas encontradas na literatura são um algoritmo heurístico específico para o problema e

outro baseado em colônia artificial de abelhas por Ghoshal e Sundar (2020) e recentemente, um algoritmo genético de estado fixo por Ghoshal e Sundar (2023).

Em relação aos problemas da segunda classe, Cerrone e Russo (2023) propuseram um algoritmo de Dijkstra com rotulação limitada como abordagem heurística para o problema do menor caminho com k rótulos (PMCkR). Eles também propuseram um método exato usando programação linear inteira mista (PLIM), melhorado por técnicas de redução do grafo. A primeira definição formal do PMCkR foi apresentada por Ferone, Festa e Pastore (2019). Eles propuseram uma formulação matemática com *branch and bound* e, em um trabalho posterior, um algoritmo de programação dinâmica (FERONE et al., 2021).

Broersma et al. (2005) demonstraram que a versão de decisão do PMCkR é NP-completa, por redução do problema 3-SAT. Como consequência, o problema do menor caminho colorido (PMCC) é NP-difícil. No caso de rótulos com pesos, Hassin, Monnot e Segev (2007) propuseram algoritmos aproximativos para o PMCC e para o PAGRM, que também é NP-difícil. Neste caso, eles focaram em minimizar o custo total dos rótulos, em vez do número de rótulos.

O PAGRM, introduzido por Chang e Leu (1997), é um dos problemas mais discutidos no contexto de GARs. Uma definição equivalente para o problema consiste em encontrar um subconjunto mínimo de rótulos tal que o subgrafo induzido seja conexo. Entre as soluções propostas, Silva et al. (2019a) apresentaram uma versão revisada do algoritmo da cobertura máxima de vértices e uma nova meta-heurística baseada em PLIM. A versão generalizada do PAGRM, que permite múltiplos rótulos em cada aresta, foi discutida por Silva et al. (2019b). Eles propuseram um modelo de PLI a analisaram o politopo associado.

Outro problema do terceiro tipo é o problema do corte mínimo colorido (PCMC), que busca um corte *s-t* usando o menor número de rótulos. Ele foi discutido por Bordini et al. (2019), que propuseram duas abordagens baseadas na heurística VNS. O PCGRM, que busca qualquer corte, em vez de apenas um corte *s-t*, foi abordado por Silva et al. (2018) com métodos heurísticos usando conceitos de algoritmos genéticos e VNS.

Por fim, Granata et al. (2013) discutem o PFMRM, demonstrando que é NP-difícil e propondo duas soluções exatas, uma baseada em fluxo e um melhoria considerando as informações de vértices e rótulos necessários. Essa formulação é detalhada na Subseção 2.2.1, adaptada para o PFRM, considerando a informação do valor desejado de fluxo.

Granata et al. (2013) também propõem uma solução para o PFMRM baseada na meta-heurística SVNS, uma variante do VNS que aceita soluções próximas à melhor conhecida, mas não necessariamente melhores, objetivando explorar melhor soluções distantes da incumbente. Para o SVNS, são definidas estruturas de vizinhança baseadas na quantidade de rótulos a serem removidos e adicionados a partir da solução atual. A busca local, por sua vez, examina até 200 vizinhos na vizinhança ativa até encontrar uma solução melhor. Este SVNS foi utilizado como referência para análise do desempenho dos métodos propostos neste trabalho.

2.2.1 Formulação baseada em fluxo

Granata et al. (2013) propõem duas formulações baseadas em fluxo para o PFMRM. A primeira formulação deriva diretamente da definição do problema, enquanto a segunda utiliza as informações dos elementos necessários para melhorar a anterior. O programa (5) a (13) apresenta uma variação, para o PFRM, dessa segunda formulação. Esse programa é utilizado como referência para análise dos métodos exatos propostos neste trabalho.

$$|L_{\mathrm{N}}| + \min \sum_{\ell \in L \setminus L_{\mathrm{N}}} z_\ell \tag{5}$$

sujeito a

$$c_a z_\ell \geq f_a, \qquad \forall a \in A,\ \ell = \ell_a \notin L_{\mathrm{N}} \tag{6}$$

$$\sum_{a \in \delta_v^+} f_a - \sum_{a \in \delta_v^-} f_a = 0, \qquad \forall v \in V \setminus \{s,t\} \tag{7}$$

$$\sum_{a \in \delta_s^+} f_a - \sum_{a \in \delta_s^-} f_a \geq f_{\mathrm{g}} \tag{8}$$

$$\sum_{a \in \delta_t^+} f_a - \sum_{a \in \delta_t^-} f_a \leq -f_{\mathrm{g}} \tag{9}$$

$$\sum_{a \in \delta_v^+} f_a \geq f_{\mathrm{g}} - f_{v-}, \qquad \forall v \in V_{\mathrm{N}} \tag{10}$$

$$\sum_{a \in \delta_v^-} f_a \geq f_{\mathrm{g}} - f_{v-}, \qquad \forall v \in V_{\mathrm{N}} \tag{11}$$

$$z_\ell \in \{0,1\}, \qquad \forall \ell \in L \setminus L_{\mathrm{N}} \tag{12}$$

$$0 \leq f_a \leq c_a, \qquad \forall a \in A \tag{13}$$

A função objetivo (5) representa a quantidade de rótulos utilizados, a ser minimizada. Como os rótulos necessários devem estar presentes na solução, já são fixados. Para cada $\ell \in L \setminus L_{\mathrm{N}}$, a variável binária z_ℓ é igual a 1 quando o rótulo ℓ está presente na solução, ou 0, caso contrário. Esse domínio está descrito na restrição (12). Também têm-se variáveis reais f_a representando o fluxo em cada arco $a \in A$. O valor de f_a não pode ser negativo e está limitado à capacidade c_a do arco, como mostra a restrição (13). Quando o rótulo ℓ_a correspondente a um arco $a \in A$ não está presente na solução, também não deve haver fluxo nele. Nesses casos, a restrição (6) anula a capacidade dos arcos, forçando o fluxo para zero. Nos demais casos, é indiferente.

As restrições (7) a (9) garantem que as variáveis f_a representam um fluxo válido. A lei da conservação do fluxo, apresentada na Equação (4), está refletida na restrição (7), garantindo que, à exceção de s e t, o fluxo que entra em cada vértice seja igual ao que dele. A restrição (8) garante que o valor do fluxo seja pelo menos o fluxo dado, assim como a restrição (9). Note-se que não é necessário garantir que o fluxo que sai da origem é igual ao que chega ao destino, pois isso decorre da restrição (7), como discutido na Seção 2.1.

Além das informações de rótulos necessários presentes em (5), (6) e (12), o vértices necessários também refletem melhorias no modelo, como observa-se nas restrições (10) e (11). Dado um vértice necessário $v \in V_{\mathrm{N}}$, seja f_{v-} o valor do fluxo máximo no grafo $D[V \setminus \{v\}]$. Note-se que, se não for possível obter f_{g} sem o vértice v, ou seja, se $f_{v-} < f_{\mathrm{g}}$, então é necessário que o fluxo que percorre o vértice v seja pelo menos a diferença $f_{\mathrm{g}} - f_{v-}$. A restrição (10) reflete isso para o fluxo que entra no vértice, enquanto (11) refere-se ao fluxo que sai do vértice. Caso $f_{v-} \geq f_{\mathrm{g}}$, então $f_{\mathrm{g}} - f_{v-} \leq 0$ e ambas as restrições tornam-se indiferentes.

3 MÉTODOS EXATOS

Este capítulo descreve os métodos exatos propostos para o PFRM, baseados na FCC de Silva (2018). A partir dela, são propostos dois algoritmos de *branch and cut*.

3.1 Formulação baseada em cortes coloridos

Um corte é uma partição (S,T) de V tal que $s \in S, t \in T$, considerando $S \subset V, T = V \setminus S$, por ser uma partição. Por simplicidade, basta um conjunto $S \subset V$ tal que $s \in S, t \notin S$ para representar um corte. Adicionalmente, $\delta_S^+ = \{a \in A \mid a = (v,w), v \in S, w \notin S\}$ representa o conjunto dos arcos que saem S para $V \setminus S$, enquanto $L_S^+ = \left\{\ell \in L \mid \exists a \in \delta_S^+, \ell = \ell_a\right\}$ representa os rótulos presentes em algum arco de δ_S^+. A partir dessas definições, o programa (14) a (16) apresenta a FCC para o PFRM.

$$\min \sum_{\ell \in L} z_\ell \tag{14}$$

sujeito a

$$\sum_{\ell \in L_S^+} z_\ell \sum_{\substack{a \in \delta_S^+ \\ \ell = \ell_a}} c_a \geq f_{\mathrm{g}}, \qquad \forall S \subset V, s \in S, t \notin S \tag{15}$$

$$z_\ell \in \{0,1\}, \qquad \forall \ell \in L \tag{16}$$

Assim como na FF, a função objetivo (14) busca minimizar a quantidade de rótulos, com as variáveis binárias z_ℓ representando a presença ou não de um rótulo na solução. O domínio de tais variáveis é descrito na restrição (16). As restrições (15) garantem que o fluxo no grafo seja pelo menos igual ao fluxo dado. O somatório interno acumula as capacidades dos arcos de um rótulo fixo. enquanto o somatório externo acumula a soma das capacidades apenas dos rótulos presentes na solução. Então, considerando apenas o subgrafo $D[L']$, onde $L' = \{\ell \in L; z_\ell = 1\}$ é o conjunto dos rótulos presentes na solução, tem-se que a soma dos capacidades dos arcos deve ser pelo menos f_{g} em qualquer corte no subgrafo.

A base para esse conjunto de restrições reside no Teorema 1, que permite converter um modelo baseado em fluxo em um dual baseado em cortes. A análise dos cortes é realizada em termos da soma das capacidades dos arcos em todos os cortes do grafo, o que incluirá o corte mínimo. No caso específico das restrições (15), tem-se que é necessário que a soma das capacidades dos arcos com rótulo presente na solução seja pelo menos f_{g} em todos os cortes do grafo. Em particular, isso deve valer para o corte mínimo, resultando que o fluxo máximo também é pelo menos f_{g}.

A informação dos rótulos necessários pode ser utilizada para melhorar as restrições (15). A princípio, observa-se que se o somatório interno pode ser limitado a f_{g}, para evitar coeficientes muito altos. No caso de um rótulo ser necessário, a certeza de sua presença na solução permite descontar a soma das capacidades dos seus arcos do fluxo dado, pois tal fluxo já está garantido naquele corte. Com isso, pode-se substituir as restrições (15) pelas restrições (17).

$$\sum_{\ell \in L_S^+ \setminus L_{\mathrm{N}}} z_\ell \min \left(\sum_{\substack{a \in \delta_S^+ \\ \ell = \ell_a}} c_a \,,\; f_{\mathrm{g}} - \sum_{\substack{a \in \delta_S^+ \\ \ell_a \in L_{\mathrm{N}}}} c_a \right) \geq f_{\mathrm{g}} - \sum_{\substack{a \in \delta_S^+ \\ \ell_a \in L_{\mathrm{N}}}} c_a \,, \qquad \forall S \subset V, s \in S, t \notin S \tag{17}$$

A base para as restrições (17) permanece a mesma. No entanto, não são verificados todos os rótulos presentes no corte, como antes. Ao desconsiderarem-se os rótulos necessários, deve-se levar em conta sua contribuição para o fluxo. Isso é observado no lado direito da inequação. Quanto ao uso da função min, seu único papel é limitar cada coeficiente do lado esquerdo ao lado direito, de modo a evitar valores desnecessariamente altos, como já discutido. Como as inequações passam a considerar apenas os rótulos necessários, a função objetivo deve ser adaptada de forma correspondente, como apresentado na equação (5).

3.2 Algoritmos de *branch and cut*

Uma vez que o conjunto de restrições da FCC possui tamanho exponencial, torna-se impraticável adicionar todas ao modelo. Para isso, dois algoritmos *branch and cut* são propostos.

O primeiro deles é apresentado no Algoritmo 3, que inicia com um modelo sem restrições. Após a primeira execução, obtém-se uma solução inicial $S_0 \subseteq L$. Então, pode-se calcular o fluxo máximo f_0 no subgrafo $D[S_0]$. Esse fluxo gera um corte K_0 em D, cuja restrição correspondente é adicionada ao modelo, de acordo com as inequações do modelo. O processo é repetido, obtendo-se uma nova solução S_1 que gera um fluxo f_1 e um corte K_1, que corresponde a uma restrição a ser adicionada. O algoritmo encerra-se ao obter um fluxo f_i tal que $|f_i| \geq f_{\mathrm{g}}$.

Algoritmo 3: Algoritmo de *branch and cut* para a FCC

```
1 Procedimento BRANCH-AND-CUT-1(D = (V,A,L), f_g)
2     Seja P um modelo de programação linear inteira de acordo com (5) e (16) ;
3     S ← ∅ ;
4     Enquanto FLUXO-MÁXIMO(D[S]) < f_g faça
5         Seja K o corte gerado por FLUXO-MÁXIMO(D[S]) ;
6         Adicione a P a restrição (17) correspondente a K ;
7         S ← conjunto de rótulos presentes na solução do modelo P ;
8     Retorne S;
```

Uma segunda versão do método é apresentada no Algoritmo 4, que executa os mesmos passos que o primeiro algoritmo, mas em duas etapas. Inicialmente, são relaxadas as restrições

de integralidade, de modo que o resolvedor matemático trabalhe apenas na raiz da árvore de busca. Nesse contexto, as etapas descritas no primeiro algoritmo são executadas para obter-se uma solução que não é necessariamente inteira, podendo utilizar um rótulo parcialmente. Para o cálculo do fluxo máximo no subgrafo induzido, entende-se que utilizar uma fração de um rótulo equivale a reduzir a capacidade dos arcos correspondentes à mesma fração. Após a execução na raiz, as restrições de integralidade são reintroduzidas e repete-se o processo descrito no algoritmo anterior. Com isso, obtém-se uma solução inteira.

Algoritmo 4: Algoritmo de *branch and cut* para a FCC em duas etapas

```
1  Procedimento BRANCH-AND-CUT-2(D = (V,A,L), f_g)
2      Seja P um modelo de programação linear de acordo com (5) ;
3      S ← ∅ ;
4      Enquanto FLUXO-MÁXIMO(D[S]) < f_g faça
5          Seja K o corte gerado por FLUXO-MÁXIMO(D[S]) ;
6          Adicione a P a restrição (17) correspondente a K ;
7          S ← rótulos e frações correspondentes da solução do modelo P ;
8      Adicione a P as restrições de integralidade (16) ;
9      Enquanto FLUXO-MÁXIMO(D[S]) < f_g faça
10         Seja K o corte gerado por FLUXO-MÁXIMO(D[S]) ;
11         Adicione a P a restrição (17) correspondente a K ;
12         S ← conjunto de rótulos presentes na solução do modelo P ;
13     Retorne S;
```

A diferença entre os algoritmos está no fato de que o segundo algoritmo já inicia-se com várias restrições adicionadas ao modelo, obtidas durante o processamento na raiz. Isso garante uma convergência mais rápida do algoritmo. No entanto, a execução na raiz pode demorar um pouco mais a convergir. De modo geral, isso potencialmente acarreta num algoritmo mais lento, mas com uma convergência mais estável.

4 MÉTODOS HEURÍSTICOS

Este capítulo descreve os métodos heurísticos propostos para o PFRM. A Seção 4.1 descreve o GRASP reativo e seus algoritmos de construção e busca local. Na Seção 4.2, é descrito o AGM, com construção gulosa aleatorizada, elitização e recombinação por caminhos. Por fim, a Seção 4.3 apresenta o método de manutenção dinâmica de fluxo, que melhora o desempenho nos cálculos de fluxo do AGM.

4.1 GRASP reativo

A meta-heurística GRASP (do inglês, *greedy randomized adaptative search procedure*, ou procedimento de busca adaptativa gulosa aleatorizada) é um método composto por uma construção gulosa aleatorizada e uma busca na vizinhança da solução construída. Iterativamente, uma solução é construída e uma busca local é realizada, enquanto não for atingido o critério de parada. Retorna-se, então, a melhor solução encontrada.

A etapa de construção é apresentada no Algoritmo 5. Ele funciona iterativamente adicionando rótulos à solução atual S até que o fluxo máximo em $D[S]$ seja maior do que ou igual ao valor dado de fluxo f_{g}. Isso pode ser observado no laço das linhas 3–6.

Algoritmo 5: Etapa de construção gulosa aleatorizada

```
1 Procedimento CONSTRUÇÃO(D = (V,A,L), f_g, α)
2     Seja S o conjunto de rótulos necessários de D;
3     Enquanto FLUXO-MÁXIMO(D[S]) < f_g faça
4         Seja (ℓ_i) uma permutação de L \ S, não-crescente em
            FLUXO-MÁXIMO(D[S ∪ {ℓ_i}]);
5         Seja ℓ tomado aleatoriamente de {ℓ_1, ..., ℓ_k}, com k = ⌈α · |L \ S|⌉;
6         S ← S ∪ {ℓ};
7     Retorne S;
```

Na linha 2, o algoritmo define a solução inicial como o conjunto de rótulos necessários, ou seja, aqueles sem os quais é impossível obter um fluxo com valor igual ou superior a f_{g}. Para determinar esse conjunto, remove-se cada rótulo ℓ individualmente, verificando se o valor do fluxo máximo em $D[L \setminus \{\ell\}]$ é menor que f_{g}.

A parte gulosa e aleatória está na forma como é escolhido o rótulo a ser adicionado à solução atual. A linha 4 define a sequência dos candidatos, ordenados por quem maximiza o fluxo da solução. A seguir, na linha 5 escolhe-se aleatoriamente um rótulo dentre os $\alpha\%$ melhores. Observe-se que o parâmetro $\alpha \in [0,1]$ define o quão aleatória é a solução, sendo $\alpha = 0$ uma solução totalmente gulosa e $\alpha = 1$, totalmente aleatória.

Após a construção, é executada uma busca na vizinhança da solução construída. A etapa de busca local adaptada ao PFRM é apresentada no Algoritmo 6, que recebe um grafo D, um valor dado de fluxo f_g e uma solução S para o PFRM em D e tenta minimizar a quantidade de rótulos em S.

Algoritmo 6: Etapa de busca local

```
1 Procedimento BUSCA-LOCAL(D = (V,A,L), f_g, S)
2     R ← {s ∈ S; FLUXO-MÁXIMO(D[S \ {s}]) ≥ f_g};
3     Enquanto R ≠ ∅ faça
4         S ← S \ {r}, com s ∈ R tomado aleatoriamente;
5         R ← {s ∈ S; FLUXO-MÁXIMO(D[S \ {s}]) ≥ f_g};
6     Retorne S;
```

As linhas 2 e 5 do algoritmo definem o conjunto R dos rótulos de S que podem ser removidos sem diminuir o valor do fluxo máximo abaixo de f_g. Então, um desses rótulos é escolhido aleatoriamente e removido de S e o conjunto R é recalculado, como observa-se no laço das linhas 3–5. O algoritmo termina ao obter-se uma solução na qual nenhum rótulo pode ser removido sem prejuízo do valor do fluxo.

As duas etapas estão reunidas no Algoritmo 7, que descreve o GRASP reativo proposto. A estrutura básica do GRASP pode ser observada nas linhas 4–5 e 10–11, onde definimos a solução inicial e o laço principal, executamos a construção gulosa aleatorizada e a busca local e mantemos a melhor solução encontrada para ser retornada na linha 13.

Algoritmo 7: GRASP Reativo

```
1  Procedimento GRASP-REATIVO(D = (V,A,L), f_g)
2      Para i = 0,1,...,10, seja α_i = i/10;
3      Para i = 0,1,...,10, s_i ← 0, n_i ← 0, p_i ← 1/11;
4      S ← L;
5      Enquanto tempo limite não atingido faça
6          Se o número de iterações é múltiplo de 10 então
7              Para i = 0,1,...,10, seja q_i = n_i/s_i (ou a média dos q_i, se inválido);
8              Para i = 0,1,...,10, p_i ← q_i / Σ_{j=0}^{10} q_i;
9          Seja i ∈ {0,1,...,10} tomado aleatoriamente com probabilidade p_i;
10         S' ← BUSCA-LOCAL(D, f_g, CONSTRUÇÃO(D, f_g, α_i));
11         Se |S'| < |S| então S ← S';
12         s_i ← s_i + |S'|, n_i ← n_i + 1;
13     Retorne S;
```

Em relação à parte reativa, a linha 2 define os valores permitidos para α, a saber, 0–100%, com passos de 10%. A linha 3 define os valores iniciais de s_i, n_i e p_i, que representam a soma das cardinalidades e a quantidade de soluções encontradas utilizando $\alpha = \alpha_i$ e a probabilidade

de selecionar α_i, respectivamente. Inicialmente, todos os valores são equiprováveis. Sempre que uma nova solução é gerada, a linha 12 atualiza os valores de s_i e n_i correspondentes.

Por fim, temos as linhas 6–8. A cada 10 iterações do laço principal, as probabilidades são atualizadas. Para os valores de α_i que já foram utilizados, define-se q_i como o inverso da média da quantidade de rótulos nas soluções ou o valor médio, quando não houver dados anteriores, como mostra a linha 7. A linha 8, então, define a probabilidade de cada α_i ser selecionado. Note-se que p_i é proporcional a q_i que, por sua vez, é inversamente proporcional à cardinalidade média das soluções. Assim, quanto menos rótulos nas soluções geradas por determinado α_i, maior a probabilidade dele ser selecionado novamente.

4.2 Algoritmo genético melhorado

A estrutura de um algoritmo genético simples começa por gerar uma população inicial de possíveis soluções para o problema. O processo consiste, então, em gerar uma nova população através de uma fase de reprodução e repetir o processo até um critério de parada ser satisfeito. A reprodução é composta por duas operações, recombinação (ou *crossover*) e mutação. Ao final de cada iteração, é definida uma população sobrevivente, que avançará para a iteração seguinte.

Neste trabalho, é proposto um algoritmo genético melhorado. São três melhorias principais, em relação a um algoritmo genético simples. A primeira delas é construir a população utilizando a etapa de construção do GRASP, apresentada no Algoritmo 5. O tamanho da população foi definido como a quantidade de vértices no grafo, $|V|$, e o parâmetro α foi escolhido aleatoriamente para a construção de cada indivíduo, com exceção do primeiro, que utilizou $\alpha = 0$ para garantir que o caso guloso seja considerado. Também é executada uma busca local em cada elemento construído.

Outra melhoria consiste em elitizar a população, através de um sistema de três classes. A primeira classe, com tamanho $\lceil 0{,}2\,|V| \rceil$ é constituída pelas melhores soluções encontradas até o momento. A terceira classe, de mesmo tamanho, é constituída por novas soluções geradas pelo Algoritmo 5, as mais aleatórias. A segunda classe seria intermediária entre as outras, com $|V| - 2\lceil 0{,}2\,|V| \rceil \approx 0{,}6\,|V|$ elementos. Então a cada iteração do algoritmo, as soluções são ordenadas, as melhores ficam nas primeira e segunda classes e as outras são descartadas para que um novo conjunto de soluções seja gerado para a terceira classe. Na etapa de recombinação, um dos pais é selecionado da primeira classe e o outro da segunda ou terceira classes.

A última melhoria foi utilizar a recombinação por caminhos (*path-relinking*). Dadas duas soluções S_1 e S_2, o *path-relinking* forma um caminho de S_1 a S_2 e seleciona filhos a partir de elementos deste caminho. Neste caso, formamos um caminho parcial, adicionando em S_1 os rótulos em $S_2 \setminus S_1$, limitado a quatro rótulos aleatórios. O mesmo é feito em S_2, para $S_1 \setminus S_2$. Isso produz duas novas soluções, que são adicionadas à população, após uma busca local. Todas as melhorias propostas são apresentadas no Algoritmo 8.

Algoritmo 8: Algoritmo genético melhorado

```
1  Procedimento ALGORITMO-GENÉTICO(D = (V,A,L), f_g)
2      P ← POPULAÇÃO-INICIAL(D, f_g);
3      Enquanto tempo limite não atingido faça
4          Enquanto |P| < 2|V| faça
5              Seja S_1 uma solução tomada aleatoriamente de P[1 : ⌈0,2|V|⌉];
6              Seja S_2 uma solução tomada aleatoriamente de
                 P[⌈0,2|V|⌉ + 1 : |V| − ⌈0,2|V|⌉];
7              Adicione a P as soluções geradas por RECOMBINAÇÃO(D, S_1, S_2, f_g);
8          Ordene P pelas soluções com maior aptidão e remova as duplicadas;
9          Remova os elementos finais de P, em quantidade necessária para que
             |P| ≤ |V| − ⌈0,2|V|⌉;
10         Enquanto |P| < |V| faça
11             Seja α ∈ (0,1] tomado aleatoriamente;
12             Adicione BUSCA-LOCAL(D, f_g, CONSTRUÇÃO(D, f_g, α_i));
13             ao final do vetor P;
14     Retorne a solução S ∈ P com maior aptidão;
```

No algoritmo, a linha 2 constrói a população com $|V|$ elementos, ordenados pela aptidão das soluções. Então o laço principal realiza a etapa de recombinação (por *path-relinking*, nas linhas 4–7), seleciona os melhores elementos para as primeiras classes (linha 8), remove os excedentes (linha 9) e constrói novas soluções para a terceira classe (linhas 10–12). Por fim, retorna-se a melhor solução encontrada, na linha 13.

4.3 Manutenção dinâmica do fluxo

Nos métodos heurísticos propostos, há uma necessidade frequente de calcular o fluxo após inserir ou remover rótulos. Para melhorar esse processo, é proposto um método de manutenção dinâmica de fluxo a partir do algoritmo de Edmonds e Karp (1972). Deseja-se um modo eficiente de adicionar ou remover todos os arcos que compartilham um dado rótulo.

Atualização do fluxo

Para atualizar o fluxo, é utilizado o algoritmo de Edmonds e Karp (1972). Não é necessário, no entanto, sempre iniciar com um fluxo vazio. Dado um fluxo qualquer, pode-se prosseguir com o método descrito no Algoritmo 9, até o valor do fluxo ser máximo ou, pelo menos, f_g.

O Algoritmo 9 inicia com um grafo $D(V,A,L)$, uma origem s, um destino t, um fluxo inicial $f : A \to \mathbb{R}$ entre os vértices s e t e um valor dado de fluxo f_g. Ele retorna um fluxo com valor mínimo de f_g ou um fluxo máximo, caso não exista fluxo com valor maior do que ou igual ao valor dado. O laço das linhas 3–5 corresponde ao algoritmo de Edmonds e Karp (1972), que é executado a partir do grafo residual da linha 2. A linha 6 retorna o fluxo resultante.

Algoritmo 9: Cálculo do fluxo máximo a partir de um fluxo dado

```
1 Procedimento ATUALIZA-FLUXO(D = (V,A,L), s, t, f : A → ℝ, f_g)
2     Seja R o grafo residual de D, após o fluxo f;
3     Enquanto houver caminho s-t em R e |f| ≤ f_g faça
4         Seja P o caminho resultante de uma BFS entre s e t em R;
5         Sature P, atualizando o fluxo f e o grafo residual R;
6     Retorne f;
```

Para atualizar o fluxo pela primeira vez, basta definir f como um fluxo vazio. Após a adição de um rótulo, basta definir o fluxo como zero nos arcos adicionados e executar o algoritmo normalmente no novo grafo. Também pode-se utilizar o algoritmo para calcular o fluxo máximo, bastando definir um valor suficientemente alto de f_{g}.

Remoção de rótulos

Na remoção de um rótulo, é necessário retirar o fluxo que passa por arcos que possuem tal rótulo. Esse procedimento é descrito no Algoritmo 10. A estrutura é semelhante ao algoritmo anterior, mas desta vez o processo é executado no grafo de fluxo, não no residual, como pode-se observar nas linhas 2–5.

Algoritmo 10: Cálculo do fluxo máximo após a remoção de um rótulo

```
1 Procedimento REMOVE-RÓTULO(D = (V,A,L), s, t, f : A → ℝ, f_g, ℓ_rem)
2     Seja F o grafo de fluxo de D, com o fluxo f;
3     Enquanto houver arco a com rótulo ℓ_rem em F faça
4         Seja P o caminho resultante de uma BFS através de a em F ;
5         Remova o máximo de fluxo possível de P, atualizando f e F;
6     Sejam L' = L \ {ℓ_rem} e A' = {a ∈ A; ℓ_a ≠ ℓ_rem};
7     Retorne ATUALIZA-FLUXO(D[L'], s, t, f, f_g);
```

O laço das linhas 3–5 é executado enquanto ainda houver fluxo em um arco com o rótulo a ser removido. Então o objetivo é zerar o fluxo em todos os arcos que serão removidos do grafo para, em seguida, adicionar fluxo no subgrafo que não possui tais arcos, como observamos nas linhas 6–7. O Algoritmo 9 é utilizado para obter o fluxo máximo após remover parte do fluxo.

Um comentário sobre a linha 4 faz-se necessário. A BFS através de a consiste em duas etapas. Dado $a = (u, v)$, inicia-se com uma BFS a partir de v, parando ao encontrar u ou t. No primeiro caso, obtém-se um ciclo contendo a, que é retornado. No segundo caso, executa-se uma segunda BFS a partir de u até s, obtendo-se um caminho de s a t passando por a. Caso possua algum ciclo no caminho, é removido, obtendo-se um novo caminho que não necessariamente contém o arco a. Ao final, a maioria dos casos resulta em um caminho de s a t passando por a. Em alguns casos, o caminho pode ser um ciclo contendo a ou um caminho de s a t que não

possua a. Esses dois casos foram adicionados para garantir que sempre exista uma saída para o fluxo até que seja possível remover todos os arcos o rótulo a ser removido.

Em relação à implementação, convém ressaltar que os arcos não são inseridos e removidos de fato. Apenas os rótulos são marcados como visíveis ou invisíveis e, onde há referência aos arcos do grafo, são considerados apenas os arcos com rótulos visíveis. O mesmo é válido para todos os métodos e algoritmos propostos neste trabalho.

5 EXPERIMENTOS COMPUTACIONAIS

Em relação aos experimentos computacionais, a Seção 5.1 descreve os resultados dos testes com os métodos exatos. A Seção 5.2, por sua vez, descreve os experimentos com os métodos heurísticos. Os experimentos com o método de manutenção dinâmica de fluxo são descritos à parte, na Seção 5.3. Em todos os casos, as soluções foram implementados na linguagem C++, utilizando o compilador g++ versão 8.3.0-6 (opção de compilação -O3), com o auxílio do resolvedor matemático CPLEX versão 12.8, em um computador com 16 GB de RAM DDR4, processador Intel Core i7-7500U (64 bits), com CPU contendo 2 núcleos de 2,70 GHz (quatro núcleos virtuais) e frequência máxima de núcleo único de 3,50 GHz.

5.1 Métodos exatos

O conjunto de teste consistiu de grafos gerados aleatoriamente, com base em dois tamanhos (100 e 200 vértices) e três densidades (0,2, 0,5 e 0,8), utilizando quatro conjuntos de rótulos (com 25%, 50%, 100% e 125% do tamanho do grafo) e três valores de fluxos (com 50%, 75% e 100% do fluxo máximo), com as capacidades dos arcos limitadas a 10. Para cada configuração, foram geradas 10 instâncias aleatórias, totalizando 720 testes. Os resultados dos experimentos com os métodos exatos estão reunidos nas Tabelas 1 e 2.

Cada linha das tabelas corresponde às médias dos resultados em um conjunto de 30 instâncias, agrupados pelo tamanho do grafo ($|V| = 100$ na Tabela 1 e $|V| = 200$ na Tabela 2), quantidade de rótulos ($|\mathrm{L}|$) e razão entre fluxo dado e o máximo ($f_\mathrm{g}/f_\mathrm{max}$). Para cada conjunto de instâncias, podem ser observados a média das soluções ótimas (opt) e da quantidade de rótulos necessários ($|\mathrm{L_N}|$). Nos casos onde $f_\mathrm{g} = f_\mathrm{max}$, há uma quantidade expressiva de rótulos necessários em comparação com os rótulos presentes na solução. Por outro lado, são raros os rótulos necessários para $f_\mathrm{g} \in \{0{,}5f_\mathrm{max}, 0{,}75f_\mathrm{max}\}$. Isso já indica que as otimizações propostas terão efeito reduzido nesses casos, resultando num pior desempenho.

Em relação aos métodos testados, tem-se a FF e dois algoritmos de *branch and cut* (FCC1 e FCC2). Para cada um deles, é informado o tempo médio para obter a solução ótima ($\mathrm{t(s)}$) e a quantidade de nós na árvore de busca do resolvedor matemático (nós). Os valores em negrito representam os melhores resultados de cada conjunto de instâncias. Observa-se que o métodos FCC1 e FCC2 superaram o desempenho da FF na maioria dos casos; de fato, FCC1 superou a FF em 99% das instâncias, enquanto FCC2 superou a FF em 97% das instâncias.

Convém destacar que em 4 das 720 instâncias, o algoritmo FCC1 não convergiu dentro do limite de memória, devido a um número alto de nós e cortes necessários. Esses casos estão destacados na tabela como asteriscos (*). Em todos eles, o algoritmo obteve uma solução com exatamente um rótulo a mais do que o ótimo. Esse problema é resolvido pelo FCC2, que

Tabela 1 – Resultados dos experimentos com os métodos exatos em grafos de 100 vértices.

$\|\mathbf{L}\|$	$\frac{\mathbf{f_g}}{\mathbf{f_{max}}}$	**opt**	$\|\mathbf{L_N}\|$	**FF**			**FCC1**			**FCC2**				
				t (s)	**$\mathbf{t_r}$ (s)**	**nós**	**t (s)**	**nós**	**cortes**	**t (s)**	**$\mathbf{t_r}$ (s)**	**nós**	**cortes**	**$\mathbf{cortes_r}$**
	0,50	6,53	0,00	0,27	0,08	0,93	0,07	35,20	11,47	**0,05**	**0,04**	**0,53**	**0,00**	7,13
25	0,75	11,00	0,03	0,25	0,07	1,17	**0,03**	5,83	3,70	0,04	**0,03**	**0,00**	**0,00**	4,50
	1,00	19,00	18,60	0,15	0,05	**0,00**	**0,01**	**0,00**	**0,00**	**0,01**	**0,01**	**0,00**	**0,00**	0,00
	0,50	9,87	0,00	0,75	0,11	9,17	**0,07**	71,27	15,63	0,10	**0,09**	**7,87**	**0,67**	18,27
50	0,75	16,80	0,07	0,28	0,07	2,60	**0,06**	52,73	10,07	0,07	**0,06**	**0,40**	**0,00**	10,40
	1,00	29,30	26,93	0,20	0,07	0,07	**0,01**	0,10	0,47	**0,01**	**0,01**	**0,00**	**0,00**	0,63
	0,50	14,30	0,00	34,00	**0,16**	**825,33**	14,29	11406,77	416,03	**1,49**	0,17	2341,60	**63,80**	46,13
100	0,75	24,47	0,00	1,18	**0,10**	**30,40**	0,21	678,93	37,10	**0,13**	0,12	40,57	**1,20**	27,00
	1,00	42,77	33,50	0,26	0,08	**1,03**	**0,01**	3,60	1,60	0,02	**0,01**	1,90	**0,63**	2,00
	0,50	15,77	0,00	80,05	**0,20**	**2006,03**	***43,60*	29106,23	595,70	**24,19**	0,22	18523,90	**186,33**	61,73
125	0,75	27,70	0,00	38,25	**0,12**	963,07	**35,68*	15671,97	188,30	**0,40**	0,15	**692,57**	**10,53**	40,10
	1,00	48,53	35,47	0,35	0,08	**3,77**	0,04	51,90	7,17	**0,03**	**0,02**	5,37	**0,60**	5,23

Tabela 2 – Resultados dos experimentos com os métodos exatos em grafos de 200 vértices.

$\|\mathbf{L}\|$	$\frac{\mathbf{f_g}}{\mathbf{f_{max}}}$	**opt**	$\|\mathbf{L_N}\|$	**FF**			**FCC1**			**FCC2**				
				t (s)	$\mathbf{t_r}$ (s)	**nós**	**t** (s)	**nós**	**cortes**	**t** (s)	$\mathbf{t_r}$ (s)	**nós**	**cortes**	$\mathbf{cortes_r}$
50	0,50	12,03	0,00	2,21	**0,44**	1,77	**0,26**	7,53	5,37	0,52	0,46	**0,40**	**0,00**	8,50
	0,75	20,57	0,00	1,61	0,36	1,40	**0,22**	7,70	3,80	0,32	**0,26**	**0,00**	**0,00**	4,33
	1,00	36,93	36,53	1,17	0,45	**0,00**	**0,05**	0,00	0,03	0,10	**0,04**	**0,00**	**0,00**	0,03
100	0,50	18,23	0,00	3,53	**0,83**	4,53	**0,36**	31,00	8,57	0,94	0,88	**3,43**	**0,03**	20,00
	0,75	31,70	0,00	2,16	**0,53**	2,73	**0,24**	13,47	4,20	0,69	0,63	**0,00**	**0,03**	12,97
	1,00	57,10	52,63	1,43	0,55	**0,00**	**0,05**	**0,00**	0,07	0,08	**0,03**	**0,00**	**0,00**	0,10
200	0,50	26,30	0,00	42,19	**1,46**	**107,53**	**0,77**	423,50	40,27	2,20	2,04	134,30	**3,10**	55,23
	0,75	47,23	0,00	6,78	**0,81**	45,50	**0,41**	161,83	13,50	1,35	1,29	**20,60**	**0,43**	32,63
	1,00	85,47	67,13	4,17	0,65	**0,00**	**0,05**	0,33	0,47	0,07	**0,01**	**0,00**	**0,00**	0,77
250	0,50	29,43	0,00	155,21	**1,84**	**442,27**	**60,29*	27965,97	474,83	**4,46**	2,95	2328,97	**19,87**	82,83
	0,75	52,70	0,00	20,35	**1,03**	**91,77**	**0,60**	639,13	18,33	1,98	1,88	98,20	**1,03**	52,07
	1,00	93,93	69,70	1,99	0,55	**1,00**	**0,07**	6,13	1,27	0,09	**0,02**	2,13	**0,27**	2,17

convergiu em todas as instâncias. O desempenho do FCC1 superou o do FCC2 em 82% dos testes. No entanto, pode-se afirmar que o FCC2 foi mais estável em termos de convergência.

Em relação às demais colunas, são apresentadas as informações do tempo de execução na raiz da árvore de busca ($t_r\,(s)$), a quantidade de cortes adicionados pelo algoritmo (cortes) e a quantidade de cortes adicionados na raiz ($cortes_r$), quando aplicáveis. Um ponto a ser observado é em relação às quantidades de cortes. O FCC2 adiciona cortes na raiz e, na maioria dos testes, isso dispensa a necessidade de mais cortes ao restaurar as restrições de integralidade. Em comparação com o FCC1, isso resulta em menos cortes adicionados no FCC2, na maioria dos casos, mesmo somando com os cortes na raiz.

5.2 Métodos heurísticos

Esta seção apresenta os resultados dos experimentos realizados com os métodos heurísticos. O conjunto de teste consistiu em grafos gerados aleatoriamente, com base em três tamanhos (50, 100 e 200 vértices) e três densidades (0,2, 0,5 e 0,8), utilizando quatro conjuntos de rótulos (com 25%, 50%, 100% e 125% do tamanho do grafo) e cinco limites para as capacidades dos arcos (1, 5, 10, 25 e 50). Cada configuração contém 10 instâncias, totalizando 1800 testes.

As melhorias propostas para os métodos heurísticos não dependem da informação dos rótulos e vértices necessários, como no caso dos exatos. Sendo assim, não seria esperada a mesma variação entre os valores esperados de fluxo e optou-se por realizar os experimentos considerando apenas um fluxo máximo. Em vez disso, foram considerados outros testes com diferentes capacidades máximas nos arcos.

A Tabela 3 apresenta os resultados para instâncias onde $|V| = 50$. Cada linha da tabela corresponde às médias dos resultados num conjunto de 50 instâncias, agrupadas pelo tamanho do grafo ($|V|$), quantidade de rótulos ($|L|$) e densidade (d), que observou-se ter mais influência no desempenho do que a capacidade máxima. Cada instância possui um tempo limite, cuja média é apresentada na coluna $t_L\,(s)$. Então seguem os dados do SVNS, do AGM e do GRASP reativo (ou simplesmente GRASP). Cada um dos métodos apresenta a média das soluções obtidas (avg) e o tempo de obtenção da melhor solução ($t_T\,(s)$).

O tempo limite foi definido como o tempo de execução do SVNS. Sendo assim, a comparação é realizada através do tempo necessário para o algoritmo alcançar a melhor solução. o *time-to-target*. Nesse sentido, para $|V| = 50$ e $|L| \in \{12, 25\}$, os três algoritmos obtiveram as mesmas soluções, com o AGM e o GRASP obtendo a solução mais rapidamente que o SVNS. Para $|V| = 50$ e $|L| \in \{50, 62\}$, o AGM obtém soluções iguais ou melhores que os outros, com um tempo um pouco pior apenas nos grafos densos. Com mais rótulos, o GRASP tomou mais tempo e obteve soluções piores que os outros.

Com $|V| = 100$, pode-se observar na Tabela 4 que há casos onde o AGM obtém soluções melhores e piores que o SVNS, com 0,4% mais rótulos no pior caso e 1,4% menos rótulos no

Tabela 3 – Resultados dos experimentos com os métodos heurísticos em grafos de 50 vértices.

$\|V\|$	$\|L\|$	d	t_L (s)	SVNS		AGM		GRASP	
				avg	t_T (s)	avg	t_T (s)	avg	t_T (s)
50	12	0,2	1,749	**5,48**	0,055	**5,48**	0,002	**5,48**	**0,001**
		0,5	0,065	**9,60**	0,055	**9,60**	**0,002**	**9,60**	**0,002**
		0,8	2,038	**11,56**	0,065	**11,56**	0,006	**11,56**	**0,005**
	25	0,2	4,174	**7,40**	0,119	**7,40**	**0,010**	**7,40**	**0,010**
		0,5	1,682	**14,98**	0,166	**14,98**	**0,015**	**14,98**	0,021
		0,8	3,161	**21,38**	0,087	**21,38**	0,021	**21,38**	**0,018**
	50	0,2	4,336	9,40	0,289	**9,30**	**0,196**	9,38	0,400
		0,5	4,022	19,68	0,168	**19,66**	**0,114**	19,70	0,186
		0,8	5,036	**33,10**	**0,197**	**33,10**	0,644	33,24	0,751
	62	0,2	4,349	10,32	0,389	**10,18**	**0,114**	10,22	0,406
		0,5	3,744	21,38	0,182	**21,36**	0,208	21,44	**0,159**
		0,8	5,024	36,38	**0,327**	**36,36**	0,936	36,58	0,693

melhor caso. Em média, ele alcança uma solução 0,2% melhor. Quanto ao GRASP, este obtém uma solução 0,1% melhor que o SVNS em média, variando entre -0,3% e 0,7%. Observa-se, ainda, que tanto o AGM quanto o GRASP são mais rápidos que o SVNS, melhorando o tempo em 58,2% e 52,7%, respectivamente, em média.

Tabela 4 – Resultados dos experimentos com os métodos heurísticos em grafos de 100 vértices.

$\|V\|$	$\|L\|$	d	t_L (s)	SVNS		AGM		GRASP	
				avg	t_T (s)	avg	t_T (s)	avg	t_T (s)
100	25	0,2	4,275	**12,16**	0,482	**12,16**	0,041	**12,16**	**0,022**
		0,5	1,278	**20,90**	0,698	**20,90**	0,034	**20,90**	**0,029**
		0,8	4,877	**24,34**	0,682	**24,34**	0,079	**24,34**	**0,072**
	50	0,2	6,946	**14,72**	0,909	**14,72**	**0,189**	**14,72**	0,241
		0,5	7,619	**31,72**	1,319	31,78	**0,623**	**31,72**	0,828
		0,8	10,210	**42,30**	1,336	42,32	0,932	42,34	**0,801**
	100	0,2	9,642	18,88	1,698	**18,76**	**0,703**	18,84	0,815
		0,5	10,138	44,72	2,666	**44,64**	**1,234**	44,68	1,716
		0,8	10,391	**67,62**	3,052	67,88	**2,491**	67,82	3,524
	125	0,2	10,031	20,30	2,387	**20,02**	1,122	20,16	**0,762**
		0,5	10,315	49,40	3,462	**49,04**	**1,977**	49,12	1,998
		0,8	10,206	**78,16**	4,100	78,24	**2,669**	78,38	3,351

Com $|V| = 200$, então, observa-se na Tabela 5 que o AGM obteve o melhor resultado médio em todos os casos, com uma solução 8,3% menor que a solução do SVNS, em média. Já o GRASP apresenta resultados melhores que o SVNS à medida que a quantidade de rótulos aumenta, variando de uma solução 0,1% pior nos menores casos até 29,1% melhor nos maiores,

com uma solução 8,2% melhor, em média. Em relação ao tempo, o AGM obteve um tempo 80,1% melhor que o SVNS, e o GRASP obteve 79,2%, em média.

Tabela 5 – Resultados dos experimentos com os métodos heurísticos em grafos de 200 vértices.

$\|V\|$	$\|L\|$	d	t_L (s)	SVNS		AGM		GRASP	
				avg	t_T (s)	avg	t_T (s)	avg	t_T (s)
200	50	0,2	19,619	**21,24**	7,261	**21,24**	**0,236**	21,26	0,373
		0,5	14,755	**41,14**	10,068	**41,14**	**0,442**	41,16	0,532
		0,8	17,509	**48,26**	11,277	**48,26**	0,983	**48,26**	**0,951**
	100	0,2	20,625	27,12	10,399	**27,02**	**1,503**	27,04	1,515
		0,5	20,239	63,04	15,666	**62,78**	3,047	62,92	**2,418**
		0,8	20,378	86,18	16,089	**84,60**	**5,492**	84,78	6,884
	200	0,2	20,399	34,04	14,521	**33,12**	1,847	33,18	**1,427**
		0,5	20,116	115,60	19,109	**90,62**	4,732	90,64	**4,284**
		0,8	21,036	163,32	20,158	**136,38**	**7,298**	136,50	8,778
	250	0,2	20,268	36,66	15,041	**34,80**	**2,168**	34,84	2,568
		0,5	20,478	139,60	19,468	**98,94**	5,134	99,02	**4,928**
		0,8	22,040	197,30	21,514	**153,70**	8,652	153,76	**8,626**

5.3 Manutenção dinâmica de fluxo

Para o método de manutenção dinâmica de fluxo, foi utilizada uma base de teste composta por 1000 instâncias geradas aleatoriamente, Os grafos possuem quatro tamanhos: 100, 200, 500 e 1000 vértices, Para cada tamanho, há cinco grupos de instâncias: $0{,}25\,|V|$, $0{,}5\,|V|$ e $|V|$ rótulos, com valor desejado de fluxo igual a $f_{\max}$, além de instâncias com $|V|$ rótulos e de fluxos dados de $0{,}75 f_{\max}$ e $0{,}5 f_{\max}$. Para cada tamanho e grupo, há cinco grupos adicionais: densidades de 0,2, 0,5 e 0,8 sem arcos paralelos; densidade de 0,8 com $0{,}125\,|A|$ arcos paralelos, limitados a 2 arcos entre cada par de vértices; e densidade de 0,8 com $0{,}25\,|A|$ arcos paralelos, limitados a 5 arcos entre cada par de vértices. Cada uma das 100 configurações possui 10 instâncias.

São comparados dois métodos: o algoritmo de Edmonds e Karp (1972) (EK) e o método de manutenção dinâmica de fluxo proposto neste trabalho (EKD). Cada teste consistiu em executar o algoritmo com todos os rótulos presentes para observar o desempenho inicial (init). Em seguida, foram removidos todos os rótulos, um a um, para o cálculo do fluxo entre as remoções (Remoção de rótulo). Por fim, os rótulos foram readicionados, um a um, para o cálculo do fluxo após as adições (Adição de rótulo). Os resultados foram agrupados em quatro tabelas, para análise do efeito das dimensões da instância sobre o desempenho do método.

A Tabela 6 apresenta os resultados agrupados pela quantidade de rótulos. Cada linha contém o desempenho médio de 50 instâncias, totalizando 600 das 1000 instâncias totais. Neste caso, foram consideradas apenas as instâncias com $f_g = f_{\max}$. Pode-se observar que o

desempenho inicial (init) de ambas as soluções foi semelhante, o que espera-se em todos os testes, já que o algoritmo de adição dinâmica de rótulo é o mesmo que uma iteração adicional do EK partindo do fluxo anterior. Pelo mesmo motivo, o tempo médio de adição de rótulo em ambos os casos foi bem semelhante e pequeno. Em relação a remoção de rótulos, o EKD obteve um desempenho entre 54% e 89% melhor.

A Tabela 7 apresenta os resultados agrupados pelo valor desejado de fluxo, ou melhor, pela razão entre este e o fluxo máximo. Outra vez, cada linha contém a média de 50 instâncias, totalizando as 600, das 1000 instâncias, que quantidade de rótulos $|L| = |V|$. Observa-se que o EKD apresenta um desempenho entre 48% e 88% melhor que o EK na remoção de rótulos. Observa-se também que o valor dado de fluxo afeta inversamente o desempenho do algoritmo. Nas instâncias com fluxo dado maior, há mais arcos com fluxo não nulo e, consequentemente, mais arcos a serem removidos e mais caminhos por onde a remoção pode ser realizada.

A Tabela 8 apresenta os resultados agrupados pela densidade, considerando as 600 instâncias sem arcos paralelos, em grupos de 50 por linha. Neste caso, pode-se observar claramente que o desempenho do EKD é melhor nas instâncias com maior densidade, enquanto essa dimensão não afeta o EK. Podemos concluir que esse é o principal fator para que o EKD alcance desempenho mais de 80% superior ao EK, intensificando à medida que o grafo cresce.

Por fim, a Tabela 9 agrupa os resultados pelo número de arcos paralelos. Neste caso, foram consideradas 600 instâncias com densidade 0,8. A densidade paralela (d_P) apresentada corresponde à razão de arcos adicionais em comparação com a quantidade de arcos do grafo, ou seja, $d_P = 0{,}25$ significa que o grafo tem 25% mais arcos que uma instância semelhante sem arcos paralelos. Esses arcos estão distribuídos de modo que o número máximo de arcos paralelos (n_P) seja respeitado. O desempenho do EKD foi entre 60% e 84% superior ao do EK, mostrando que o EKD comporta-se melhor com multigrafos.

5.4 Discussão

Este capítulo apresentou os experimentos computacionais realizados com os métodos exatos e heurísticos, além dos experimentos específicos realizados com o método de manutenção dinâmica de fluxo. Comparando as Tabelas 1 a 5, é possível observar que ainda há uma baixa discrepância de desempenho entre os métodos exatos e heurísticos, que deve ser evidenciada com testes em grafos maiores.

Analisando apenas os métodos exatos, conclui-se que a FCC proposta, com seus dois algoritmos de *branch-and-cut*, obteve um bom resultado em comparação com a FF encontrada na literatura. Ainda pode-se destacar que o desempenho da FF diminui consideravelmente à medida que o valor esperado de fluxo diminui, evidenciando o papel dos rótulos e vértices necessários para o desempenho do método.

Em relação aos métodos heurísticos, pode-se concluir a partir das Tabela 3 a 5 que

Tabela 6 – Resultados dos métodos de fluxo (em ms), agrupados por $|L|$, com $f_g = f_{max}$.

$\|V\|$	$\frac{\|L\|}{\|V\|}$	Edmonds-Karp						Edmonds-Karp Dinâmico					
		init	Remoção de rótulo			Adição de rótulo			init	Remoção de rótulo			Adição de rótulo		
			avg	min	max	avg	min	max		avg	min	max	avg	min	max
100	0,25	0,31	0,28	0,04	0,68	0,05	0,02	0,45	**0,31**	**0,11**	0,03	0,98	**0,05**	0,00	0,46
	0,50	**0,71**	0,54	0,08	1,58	0,07	0,03	0,97	0,84	**0,25**	0,03	1,48	**0,07**	0,03	0,99
	1,00	**0,99**	0,78	0,04	1,88	0,09	0,03	1,86	1,09	**0,38**	0,03	2,19	**0,08**	0,03	1,14
200	0,25	**1,23**	1,02	0,06	10,43	0,09	0,03	5,22	1,56	**0,27**	0,03	2,40	**0,08**	0,02	1,86
	0,50	**3,75**	2,36	0,07	7,10	0,15	0,02	10,35	4,22	**0,79**	0,04	7,39	**0,14**	0,03	2,61
	1,00	**5,68**	3,87	0,06	15,87	0,18	0,03	5,90	6,19	**1,39**	0,05	13,14	**0,18**	0,03	6,62
500	0,25	10,83	8,29	0,07	33,13	0,32	0,00	26,77	**9,75**	**1,20**	0,05	18,13	**0,26**	0,03	13,94
	0,50	36,65	21,29	0,09	102,15	0,56	0,00	30,91	**33,14**	**4,96**	0,09	57,17	**0,50**	0,03	24,97
	1,00	69,10	41,22	0,08	161,16	0,78	0,03	36,42	**66,23**	**11,43**	0,12	132,07	**0,74**	0,03	27,79
1000	0,25	58,49	46,23	0,08	243,11	1,07	0,03	149,78	**43,36**	**5,06**	0,08	130,64	**0,92**	0,00	120,20
	0,50	**237,40**	142,09	0,10	795,50	1,98	0,00	231,85	238,42	**27,34**	0,18	497,87	**1,80**	0,03	145,35
	1,00	547,17	289,79	0,09	1124,88	2,56	0,04	195,09	**543,82**	**63,10**	0,26	949,80	**2,44**	0,04	146,22

Tabela 7 – Resultados dos métodos de fluxo (em ms), agrupados pelo fluxo dado, com $|L| = |V|$.

$\|\mathbf{V}\|$	$\frac{\mathbf{f_g}}{\mathbf{f_{max}}}$	**Edmonds-Karp**						**Edmonds-Karp Dinâmico**					
		init	**Remoção de rótulo**			**Adição de rótulo**			**init**	**Remoção de rótulo**			**Adição de rótulo**		
			avg	**min**	**max**	**avg**	**min**	**max**		**avg**	**min**	**max**	**avg**	**min**	**max**
100	1,00	**0,99**	0,78	0,04	1,88	0,09	0,03	1,86	1,09	**0,38**	0,03	2,19	**0,08**	0,03	1,14
	0,75	**1,01**	0,79	0,04	4,74	0,09	0,03	1,34	1,11	**0,41**	0,03	2,28	**0,08**	0,02	1,35
	0,50	**0,29**	0,28	0,04	0,72	0,05	0,03	0,54	0,29	**0,11**	0,03	0,89	**0,05**	0,03	0,58
200	1,00	**5,68**	3,87	0,06	15,87	0,18	0,03	5,90	6,19	**1,39**	0,05	13,14	**0,18**	0,03	6,62
	0,75	**5,85**	3,97	0,07	19,27	0,19	0,03	3,92	6,32	**1,52**	0,04	10,61	**0,18**	0,03	3,42
	0,50	**1,22**	1,05	0,06	2,76	0,08	0,01	1,71	1,54	**0,30**	0,03	3,03	**0,08**	0,03	1,84
500	1,00	69,10	41,22	0,08	161,16	0,78	0,03	36,42	**66,23**	**11,43**	0,12	132,07	**0,74**	0,03	27,79
	0,75	77,07	43,47	0,07	623,43	**0,83**	0,03	45,23	**68,05**	**13,39**	0,13	230,00	0,87	0,03	214,89
	0,50	11,37	8,68	0,08	44,58	0,30	0,00	24,63	**9,64**	**1,49**	0,05	28,37	**0,24**	0,00	18,54
1000	1,00	547,17	289,79	0,09	1124,88	2,56	0,04	195,09	**543,82**	**63,10**	0,26	949,80	**2,44**	0,04	146,22
	0,75	**562,87**	313,69	0,10	1357,20	2,70	0,04	226,75	567,41	**77,17**	0,27	1087,14	**2,57**	0,03	169,10
	0,50	59,62	46,03	0,08	335,32	1,08	0,03	176,18	**48,24**	**5,62**	0,10	134,00	**0,91**	0,03	110,30

Tabela 8 – Resultados dos métodos de fluxo (em ms), por densidade, sem arcos paralelos.

|V|	d	Edmonds-Karp						Edmonds-Karp Dinâmico					
		init	Remoção de rótulo			Adição de rótulo			init	Remoção de rótulo			Adição de rótulo		
			avg	min	max	avg	min	max		avg	min	max	avg	min	max
100	0,2	**0,84**	**0,58**	0,04	1,89	0,14	0,03	0,93	0,92	0,58	0,03	2,28	**0,13**	0,03	0,93
	0,5	**0,89**	0,59	0,04	1,86	0,09	0,03	1,35	0,95	**0,36**	0,03	2,22	**0,09**	0,00	1,35
	0,8	**0,85**	0,59	0,04	2,00	0,06	0,02	0,75	0,92	**0,22**	0,03	1,72	**0,06**	0,03	0,75
200	0,2	**4,58**	2,65	0,06	9,07	0,29	0,03	3,68	5,00	**2,14**	0,04	11,51	**0,28**	0,03	3,68
	0,5	**4,54**	2,80	0,06	19,27	0,19	0,01	6,62	5,02	**1,33**	0,03	13,14	**0,18**	0,03	6,62
	0,8	**4,87**	2,82	0,06	15,87	0,14	0,03	2,61	5,24	**0,77**	0,03	8,01	**0,13**	0,02	2,61
500	0,2	54,18	28,57	0,09	623,43	**1,26**	0,03	214,89	**49,74**	**17,45**	0,12	230,00	1,41	0,03	214,89
	0,5	54,97	27,54	0,08	181,88	0,83	0,00	25,70	**49,25**	**10,21**	0,07	124,42	**0,76**	0,03	25,70
	0,8	52,71	27,74	0,08	159,50	0,52	0,00	28,63	**49,77**	**5,73**	0,05	100,41	**0,48**	0,00	28,63
1000	0,2	**404,74**	197,77	0,09	1357,20	4,96	0,03	169,10	406,43	**104,61**	0,28	1087,14	**4,53**	0,02	169,10
	0,5	390,74	193,00	0,08	821,98	2,93	0,03	145,35	**383,22**	**57,41**	0,16	860,89	**2,76**	0,03	145,35
	0,8	388,78	197,01	0,10	940,87	1,58	0,00	131,13	**385,02**	**30,87**	0,08	802,38	**1,47**	0,00	131,13

Tabela 9 – Resultados dos métodos de fluxo (em ms), agrupados por arcos paralelos, com $d = 0{,}8$.

|V|	d_P	n_P	Edmonds-Karp						Edmonds-Karp Dinâmico					
			init	Remoção de rótulo			Adição de rótulo			Remoção de rótulo			Adição de rótulo	
				avg	min	max	avg	min	max	avg	min	max	avg	min	max
100	0,00	1	**0,85**	0,59	0,04	2,00	0,06	0,02	0,75	0,92	**0,22**	0,03	1,72	**0,06**	0,03	0,75
	0,12	2	**0,49**	0,54	0,04	4,74	0,06	0,03	0,83	0,57	**0,21**	0,03	1,75	**0,06**	0,03	0,83
	0,25	5	**0,25**	0,43	0,05	1,30	0,06	0,03	1,14	0,29	**0,17**	0,03	1,49	**0,06**	0,02	1,14
200	0,00	1	**4,87**	2,82	0,06	15,87	0,14	0,03	2,61	5,24	**0,77**	0,03	8,01	**0,13**	0,02	2,61
	0,12	2	**2,56**	2,42	0,07	7,32	0,11	0,03	3,41	3,04	**0,67**	0,03	8,03	**0,11**	0,02	3,41
	0,25	5	**1,18**	1,91	0,06	6,00	0,11	0,02	3,29	1,53	**0,56**	0,03	6,95	**0,10**	0,03	3,29
500	0,00	1	52,71	27,74	0,08	159,50	0,52	0,00	28,63	**49,77**	**5,73**	0,05	100,41	**0,48**	0,00	28,63
	0,12	2	30,79	25,42	0,07	125,39	0,47	0,03	25,55	**27,31**	**5,24**	0,05	92,05	**0,42**	0,00	25,55
	0,25	5	12,37	18,15	0,07	65,92	0,37	0,00	38,27	**10,74**	**3,92**	0,05	83,18	**0,33**	0,03	38,27
1000	0,00	1	388,78	197,01	0,10	940,87	1,58	0,00	131,13	**385,02**	**30,87**	0,08	802,38	**1,47**	0,00	131,13
	0,12	2	203,96	167,80	0,08	795,50	1,46	0,03	146,22	**197,37**	**27,58**	0,09	614,30	**1,33**	0,02	146,22
	0,25	5	77,32	117,61	0,08	525,87	1,31	0,03	142,61	**69,20**	**20,41**	0,10	576,71	**1,16**	0,03	142,61

ambos os métodos propostos obtiveram resultados satisfatórios quando comparados ao SVNS encontrado na literatura. Observa-se ainda que o AGM obteve desempenho superior ao GRASP. Um dos fatores que pode ter contribuído com isso é a solução puramente gulosa, que é sempre considerada no AGM, podendo ou não ser considerada no GRASP. Outro fator é a etapa de recombinação do AGM, que permite tentar aprimorar as melhores soluções já encontradas, enquanto o GRASP depende unicamente da construção gulosa aleatorizada e da busca local.

Por fim, os experimentos computacionais realizados com o método de manutenção dinâmica de fluxo evidenciaram seu potencial para os métodos heurísticos e outras aplicações com grafos dinâmicos. Observa-se nas Tabelas 6 a 9 que o melhor desempenho do método ocorre nos grafos mais densos ou com arcos paralelos, casos nos quais o algoritmo tradicional torna-se consideravelmente mais lento. Ainda assim, em todos os casos o método de manutenção iguala ou supera o desempenho do algoritmo de fluxo tradicional.

6 CONSIDERAÇÕES FINAIS

O PFRM é uma generalização do PFMRM, ambos NP-difíceis, com grande potencial de aplicação. Embora seja possível reduzir o PFRM ao PFMRM, as otimizações dos métodos exatos presentes na literatura são perdidas, pois baseiam-se nas informações de elementos necessários. Sendo assim, o estudo do PFRM é relevante e exige novas estratégias.

Como resultados desta pesquisa tem-se os métodos exatos e heurísticos propostos para o PFRM, assim como os experimentos computacionais realizados. Os resultados dos métodos exatos foram reunidos em um artigo, publicado e apresentado no LIII Simpósio Brasileiro de Pesquisa Operacional (SBPO 2021). Quanto aos métodos heurísticos e de manutenção de fluxo, um artigo foi publicado e apresentado no LII Simpósio Brasileiro de Pesquisa Operacional (SBPO 2020) com os resultados dos métodos para o PFMRM.

Sob a ótica exata, foi proposta uma formulação baseada em cortes coloridos e dois algoritmos de *branch and cut* para lidar com o número exponencial de restrições da formulação. Os experimentos computacionais demonstraram que ambos são eficientes em instâncias de tamanho moderado. O primeiro deles possuiu um desempenho melhor, mas não convergiu, em alguns casos, dentro do limite de memória estabelecido. O segundo algoritmo demonstrou maior estabilidade, ao iniciar a execução relaxando as restrições de integralidade para depois adicioná-las novamente ao modelo, o que custou um pouco do seu desempenho.

Os métodos exatos propostos concentraram-se em melhorar o tempo de construção do sistema de inequações. Como proposta para trabalhos futuros, pode-se buscar a melhoria das inequações para reduzir o tempo de resolução do sistema, como métodos de convexificação das inequações. Outra proposta consiste em abordar o problema sob a ótica de algoritmos aproximativos, ainda não presentes na literatura para o PFRM ou PFMRM.

Em relação às abordagens heurísticas, foram propostos um GRASP reativo, que ajusta automaticamente o parâmetro de aleatoriedade na construção da solução, e um algoritmo genético melhorado com construção gulosa aleatorizada baseada em GRASP, elitização e recombinação por caminhos. Ambos os métodos tiveram um desempenho semelhante e superior à abordagem encontrada na literatura, observando-se uma melhoria de cerca de 80% no *time-to-target* em relação ao SVNS, na média dos grafos com 200 vértices. Além disso, também mostraram-se capazes de encontrar melhores soluções, com destaque para o algoritmo genético, que alcançou soluções iguais ou melhores que os outros algoritmos na maioria dos casos analisados.

Também foi proposto um método de manutenção dinâmica de fluxo baseado no algoritmo de Edmonds e Karp (1972), para adaptar um fluxo à adições e remoções de rótulos, em vez de sempre recalcular o fluxo do zero. Experimentos mais detalhados foram realizados o método, mostrando sua vantagem em relação ao algoritmo clássico de Edmonds e Karp (1972). Essa van-

tagem é acentuada em grafos com maior densidade e com arcos paralelos, que podem constituir objeto de pesquisa em trabalhos futuros, assim como uma análise teórica mais aprofundada do método, um comparativo com algoritmos de fluxo mais eficientes, como o *push-relabel*, além dos demais métodos dinâmicos presentes na literatura.

Por fim, propostas de trabalhos futuros incluem, além das já citadas, aplicar as abordagens propostas em grafos maiores para observar a evolução do desempenho das soluções, analisar o problema e soluções propostas sob a ótica de multigrafos, adicionar mais restrições ao modelo exato, propor soluções heurísticas com base em outras meta-heurísticas ou hiper-heurísticas, e utilizar soluções parciais dos métodos heurísticos para refinar o modelo exato.

REFERÊNCIAS BIBLIOGRÁFICAS

AHUJA, R. A. et al. *Network Flows: Theory, Algorithms and Applications*. New Jersey: Prentice Hall, 2013. Citado 2 vezes nas páginas 14 e 18.

ALBUQUERQUE, K. M. M. et al. Métodos exatos para o problema do fluxo com rotulação mínima. In: *Anais do LIII Simpósio Brasileiro de Pesquisa Operacional*. João Pessoa: Galoá, 2021. Citado na página 13.

ALBUQUERQUE, K. M. M.; OLIVEIRA, B. L. C.; SILVA, T. G. Meta-heurísticas aplicadas ao problema do fluxo máximo com rotulação mínima. In: *Anais do LII Simpósio Brasileiro de Pesquisa Operacional*. João Pessoa: Galoá, 2020. Citado na página 13.

BONDY, J. A.; MURTY, U. S. R. *Graph Theory with Applications*. London: Macmillan, 1976. Citado 2 vezes nas páginas 17 e 20.

BORDINI, A. et al. New algorithms for the minimum coloring cut problem. *International Transactions in Operational Research*, v. 26, n. 5, p. 1868–1883, 2019. Citado na página 25.

BROERSMA, H. et al. Paths and cycles in colored graphs. *Australas. J Comb.*, Citeseer, v. 31, p. 299–312, 2005. Citado na página 25.

CARRABS, F. et al. The rainbow spanning forest problem. *Soft Computing*, Springer, v. 22, p. 2765–2776, 2018. Citado na página 24.

CERRONE, C.; RUSSO, D. D. An exact reduction technique for the k-colour shortest path problem. *Computers & Operations Research*, v. 149, p. 106027, 2023. ISSN 0305-0548. Citado na página 25.

CHANG, R.-S.; LEU, S.-J. The minimum labeling spanning trees. *Information Processing Letters*, v. 63, n. 5, p. 277 – 282, 1997. Citado na página 25.

CHERIYAN, J.; MAHESHWARI, S. N. Analysis of preflow push algorithms for maximum network flow. In: NORI, K. V.; KUMAR, S. (Ed.). *Foundations of Software Technology and Theoretical Computer Science*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1988. p. 30–48. Citado na página 22.

DANTZIG, G. B.; FULKERSON, D. R. *On the max flow min cut theorem of networks*. Santa Monica, CA: RAND Corporation, 1955. Citado na página 20.

EDMONDS, J.; KARP, R. M. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)*, v. 19, n. 2, p. 248–264, 1972. Citado 5 vezes nas páginas 20, 21, 34, 42 e 49.

FERONE, D. et al. A dynamic programming algorithm for solving the k-color shortest path problem. *Optimization Letters*, Springer, v. 15, n. 6, p. 1973–1992, 2021. Citado na página 25.

FERONE, D.; FESTA, P.; GUERRIERO, F. The rainbow steiner tree problem. *Computers & Operations Research*, v. 139, p. 105621, 2022. ISSN 0305-0548. Citado na página 24.

FERONE, D.; FESTA, P.; PASTORE, T. The k-color shortest path problem. In: ____. *Advances in Optimization and Decision Science for Society, Services and Enterprises: ODS, Genoa, Italy, September 4-7, 2019*. Cham: Springer International Publishing, 2019. p. 367–376. ISBN 978-3-030-34960-8. Citado na página 25.

FIGUEIRÊDO, C. O. et al. Meta-heurísticas aplicadas ao problema do diâmetro em grafos com arestas rotuladas. In: *Anais do LI Simpósio Brasileiro de Pesquisa Operacional*. Limeira: Galoá, 2019. Citado na página 12.

FIGUEIRÊDO, C. O. et al. Métodos exatos para o problema do diâmetro colorido. In: *Anais do LII Simpósio Brasileiro de Pesquisa Operacional*. João Pessoa: Galoá, 2020. Citado na página 12.

FORD, L. R.; FULKERSON, D. R. Maximal flow through a network. *Canadian Journal of Mathematics*, v. 8, p. 399–404, 1956. Citado na página 20.

GHOSHAL, S.; SUNDAR, S. Two heuristics for the rainbow spanning forest problem. *European Journal of Operational Research*, v. 285, n. 3, p. 853–864, 2020. ISSN 0377-2217. Citado na página 25.

GHOSHAL, S.; SUNDAR, S. A steady-state grouping genetic algorithm for the rainbow spanning forest problem. *SN Computer Science*, v. 4, n. 4, 2023. Citado na página 25.

GOLDBERG, A. V.; TARJAN, R. E. A new approach to the maximum-flow problem. *Journal of the ACM (JACM)*, ACM New York, v. 35, n. 4, p. 921–940, 1988. Citado na página 21.

GRANATA, D. et al. Maximum flow problems and an np-complete variant on edge-labeled graphs. *Handbook of combinatorial optimization*, Springer New York, New York, NY, p. 1913–1948, 2013. Citado 7 vezes nas páginas 12, 16, 20, 22, 23, 25 e 26.

HASSIN, R.; MONNOT, J.; SEGEV, D. Approximation algorithms and hardness results for labeled connectivity problems. *Journal of Combinatorial Optimization*, Springer, v. 14, n. 4, p. 437–453, 2007. Citado na página 25.

MORENO, J.; MARTINS, S.; FROTA, Y. A new approach for the rainbow spanning forest problem. *Soft Computing*, Springer, v. 24, p. 3771–3780, 2020. Citado na página 24.

SILVA, T. G. *The minimum labeling spanning tree and related problems*. 23 – 30 p. Tese (Doutorado) — Universidade Federal Fluminense, 2018. Citado 3 vezes nas páginas 12, 16 e 28.

SILVA, T. G. et al. Efficient heuristics for the minimum labeling global cut problem. *Electronic Notes in Discrete Mathematics*, v. 66, p. 23 – 30, 2018. Citado 2 vezes nas páginas 12 e 25.

SILVA, T. G. et al. Novos ótimos para o poblema da árvore geradora com rotulação mínima. In: *Anais do XLVII Simpósio Brasileiro de Pesquisa Operacional*. Porto de Galinhas: [s.n.], 2015. p. 4214–4225. Citado na página 12.

SILVA, T. G. et al. A hybrid metaheuristic for the minimum labeling spanning tree problem. *European Journal of Operational Research*, v. 274, n. 1, p. 22–34, 2019. ISSN 0377-2217. Citado na página 25.

SILVA, T. G. et al. A polyhedral approach to the generalized minimum labeling spanning tree problem. *EURO Journal on Computational Optimization*, v. 7, n. 1, p. 47–77, 2019. ISSN 2192-4406. Citado na página 25.