UNIVERSIDADE FEDERAL FLUMINENSE

FRANCISCO GLAUBOS NUNES CLÍMACO

Incorporação de métodos exatos a uma heurística para o problema de síntese de redes a 2-caminhos

Niterói

2015

UNIVERSIDADE FEDERAL FLUMINENSE

FRANCISCO GLAUBOS NUNES CLÍMACO

Incorporação de métodos exatos a uma heurística para o problema de síntese de redes a 2-caminhos

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do Grau de Mestre em Computação. Área de concentração: Algoritmos e Otimização

Orientador:
Isabel Cristina Mello Rosseti

Co-orientador:
Luidi Gelabert Simonetti

Niterói

2015

FRANCISCO GLAUBOS NUNES CLÍMACO

Incorporação de métodos exatos a uma heurística para o problema de síntese de redes a 2-caminhos

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do Grau de Mestre em Computação. Área de concentração: Algoritmos e Otimização

BANCA EXAMINADORA

Profª. Isabel Cristina Mello Rosseti - Orientador, UFF
(Presidente)

Prof. Luidi Gelabert Simonetti - Co-orientador, UFF

Prof. Thiago Ferreira de Noronha, UFMG

Prof. Yuri Abitbol De Menezes Frota, UFF

Profª. Simone de Lima Martins, UFF

Niterói
2015

À minha mãe.

Agradecimentos

Aos meus pais Francisco e Dilma, pela base familiar e apoio incondicional durante todos os períodos. Sem a presença de vocês, a realização deste trabalho não seria possível.

Ao meu orientador de graduação, Patrick Letouzé, que contribuiu bastante para minha formação acadêmica e ingresso no curso de Mestrado da UFF.

Aos meus orientadores Isabel e Luidi que, por meio de suas particularidades, me transmitiram ensinamentos e orientações que guardarei por toda a minha vida.

Aos secretários da pós-graduação, principalmente a Teresa, que colaborou sempre que possível para minha permanência no IC.

Aos membros da banca, pelas críticas e sugestões construtivas.

Aos meus colegas de curso Marcos, Edcarllos, Alberto, Igor, Heder, Alan, Gilberto, Priscila, Renata e todos os demais do IC que, direta e indiretamente, participaram dessa conquista.

Ao CNPq, pelo apoio financeiro.

Resumo

Para o projeto de redes de comunicação, é importante possuir conexões com poucas arestas no intuito de diminuir a probabilidade de atrasos e aumentar a confiança. Neste trabalho, são propostas duas estratégias que combinam programação linear inteira (PLI) com uma heurística estado-da-arte para resolução do problema de síntese de redes a 2-caminhos (2-PNDP) [6].

Essas estratégias coletam informações durante a execução dessa heurística, e as utilizam para reduzir o espaço de soluções do modelo de PLI que representa o 2-PNDP, acelerando o processamento do resolvedor de PLI, sem perda de qualidade de solução em relação às soluções obtidas por essa heurística. Na primeira estratégia, o resolvedor de PLI é invocado logo após a execução dessa heurística e tenta melhorar a solução obtida, enquanto na segunda, o resolvedor de PLI é chamado durante o processamento dessa heurística, melhorando a qualidade de seus conjuntos elite.

No intuito de validar essas propostas, foram realizados experimentos computacionais em 25 instâncias já conhecidas da literatura [4], e os resultados indicaram o benefício dessas combinações, alcançando soluções de melhor qualidade em menor tempo de processamento para a maioria dos testes.

Palavras-chave: 2-PNDP, PLI, heurística estado-da-arte

Abstract

For communication networks design, it is important to have conections with few edges in order to reduce the probability of delays and increase confidence. In this work, we propose two strategies that combine integer linear programming (ILP) with a state-of-the-art heuristic for solving the 2-path network design problem (2-PNDP) [6].

These strategies collect information during the execution of this heuristic, and use them to reduce the solution space of the ILP model which represents the 2-PNDP, accelerating the ILP solver processing without loss of solution quality compered to the solutions obtained by this heuristic. In the first strategy, the ILP solver is invoked immediately after the execution of this heuristic and tries to improve the solution obtained, while in the second, the ILP solver is called during the processing of this heuristic, improving the quality of its elite sets.

In order to validate these proposals, computational experiments were performed in 25 instances already known in the literature [4], and the results indicated the benefit of these combinations, achieving better quality solutions in less processing time for most tests.

Keywords: 2-PNDP, ILP, state-of-the-art heuristic

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

GRASP	:	*Greedy Randomized Adaptative Search Procedures*
RC	:	Reconexão por caminhos
DM-GRASP-RC	:	GRASP híbrido com *data mining* e reconexão por caminhos
MDM-GRASP-RC	:	GRASP híbrido com *multi-data mining* e reconexão por caminhos
LRC	:	Lista restrita de candidatos (*Restricted Candidate List*)
PLI	:	Programação Linear Inteira
MPO	:	Método de pós-otimização
MCF	:	Mineração de conjunto de itens frequentes
2-PNDP	:	Problema de síntese de redes a 2-caminhos

Sumário

Capítulo 1

Introdução

Problemas de otimização combinatória são comuns em nosso cotidiano, e surgem quando se quer, por exemplo, encontrar a melhor configuração para distribuição de bens de consumo, escalonamento de tarefas, planejamento de produção, entre outros. Para o projeto de redes de comunicação, é importante possuir caminhos com poucas arestas no intuito de diminuir a probabilidade de atrasos e aumentar a confiança dos dados transmitidos [19]. Motivado por essa importância, este trabalho trata o Problema de Síntese de Redes a 2-Caminhos (2-PNDP), no qual Dahl Johannessen [6] provaram que a sua versão de decisão é um problema NP-completo.

O 2-PNDP pode ser definido como segue. Seja $G(V, E)$ um grafo conectado não direcionado, tal que V representa o conjunto de vértices e E o conjunto de arestas, cada uma com custos não-negativos. Dado um conjunto de demandas D contendo pares de vértices origem-destino, o 2-PNDP consiste em encontrar um subconjunto $E' \subseteq E$ de custo mínimo contendo um 2-caminho (um caminho com, no máximo, duas arestas) entre cada par origem-destino pertencente a D.

1.1 Formulações para o 2-PNDP

Dahl e Johannessen [6] propuseram duas formulações de programação linear inteira para o 2-PNDP: modelo caminho e modelo 2-estrela.

1.1.1 Modelo caminho

No modelo caminho, variáveis representam o 2-caminho escolhido para o atendimento de cada demanda. Seja P_d o conjunto de todos os 2-caminhos para cada par origem-

destino $d \in D$. Uma variável binária z_p é definida como $z_p = 1$ se um caminho p é escolhido a partir de um conjunto P_d, ou $z_p = 0$ caso contrário. Nesse sentido, o 2-PNDP pode ser formulado como segue:

$$min \sum_{e \in E} c_e x_e \tag{1.1}$$

Sujeito a:

$$\sum_{p \in P_d} z_p = 1, \qquad \forall d \in D \tag{1.2}$$

$$z_p \leq x_e, \qquad \forall d \in D, p \in P_d, e \in p \tag{1.3}$$

$$z_p \geq 0, \qquad \forall d \in D, p \in P_d \tag{1.4}$$

$$x_e, z_p \in \{0, 1\}, \qquad \forall d \in D, p \in P_d, e \in p \tag{1.5}$$

O número de restrições (1.2) é no máximo C (número total de 2-caminhos), e define que um 2-caminho é selecionado para cada par origem-destino $d \in D$, no qual $C = |D|$. O número de restrições (1.3) é no máximo $C(2n-3)$, em que $n = |V|$, pois com exceção do 2-caminho que une de maneira direta um par origem-destino, todos os outros 2-caminhos são formados por duas arestas. O número de restrições (1.4) é no máximo $C(n-1)$, porque para cada par origem-destino existem, no máximo, $(n-1)$ 2-caminhos. Portanto, em um grafo completo, esse modelo possui $O(n^2)$ restrições.

1.1.2 Modelo 2-estrela

Sejam s e t nós distintos em G e uma 2-*estrela* como um subconjunto de E na forma

$$T(S_1, S_2) = [s, S_2] \cup [t, S_1]$$

no qual $S_1 \cup S_2 = V, S_1 \cap S_2 = \emptyset$ e $s \in S_1, t \in S_2$. Define-se $T(S_1, S_2)$ como uma 2-estrela para a aresta $[s, t]$ e Γ_{st} como o conjunto de todas as 2-estrelas existentes entre s e t.

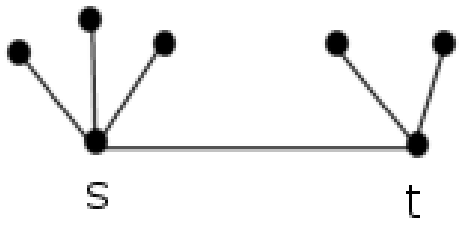


Figura 1.1: Exemplo de 2-estrela

Lema 1. *Sejam s e t nós distintos em G e seja $F \subseteq E$. Então $G(V, F)$ contém um 2-caminho entre s e t se e somente se $F \cap T \neq \emptyset$ para todo $T \in \Gamma_{st}$.*

Devido ao Lema 1, o 2-PNDP pode ser formulado como um problema de programação linear inteira (modelo 2-estrela) da seguinte maneira:

$$min \sum_{e \in E} c_e x_e \tag{1.6}$$

Sujeito a:

$$\sum_{e \in T} x_e \geq 1, \qquad \forall T \in \Gamma_{st}, \forall (s,t) \in D \tag{1.7}$$

$$x_e \in \{0, 1\}, \qquad \forall e \in E \tag{1.8}$$

no qual $x_e = 1$ indica que e pertence a um 2-caminho, e $x_e = 0$ caso contrário. As restrições (1.7) garantem que, pelo menos uma aresta de uma 2-estrela é escolhida, enquanto as restrições (1.8) garantem a integralidade da solução.

Essa formulação possui no máximo $O(n^2)$ variáveis. No caso em que o grafo G é completo, há 2^{n-2} 2-estrelas para cada par origem-destino; o número de restrições (1.7) é $C \times 2^{n-2}$; o número de variáveis inteiras é $n(n-1)/2$; e, o número de restrições do modelo 2-estrela cresce exponencialmente com n. Devido ao crescimento exponencial das restrições do modelo 2-estrela, e devido ao modelo caminho ser uma formulação mais intuitiva, esse último modelo foi escolhido para a aplicação de PLI na proposta deste trabalho.

1.2 Algoritmos propostos

O objetivo desta dissertação é desenvolver novas estratégias para a resolução do 2-PNDP, por meio do estudo das execuções da heurística estado-da-arte para esse problema integrado à utilização de métodos exatos.

Para a resolução do 2-PNDP, a maioria dos algoritmos presentes na literatura foram desenvolvidos baseados em estratégias gulosas e em técnicas de aprimoramento de soluções, incluindo-se reconexão de caminhos e mineração de dados. A primeira abordagem para resolver esse problema foi apresentada por Dahl e Johannessen [6], no qual foi proposto um algoritmo guloso que resolve, de maneira aproximada, o problema dual da relaxação linear do modelo 2-estrela, gerando uma solução primal viável.

As estratégias que serão comentadas a seguir são baseadas na metaheurística GRASP, que é uma estratégia multipartida executada iterativamente, e cada iteração é composta por duas fases: construção e busca local. Basicamente, uma solução viável é gerada de maneira gulosa e aleatória na fase de construção, e então sua vizinhança é explorada pela busca local com o objetivo de encontrar uma solução de melhor qualidade. Esse processo iterativo é repetido até que um critério de parada seja alcançado, no qual pode ser um número limite de iterações permitidas, iterações sem melhoria de solução, entre outros. Uma vez atingido esse critério de parada, a melhor solução encontrada ao longo de todas as iterações é retornada como o resultado final do processo.

O GRASP não faz uso de memória a longo prazo [7]. Nesse sentido, para resolução do 2-PNDP, Rosseti [22] propôs o método GRASP-RC de melhoria de soluções por meio do uso da metaheurística GRASP juntamente com a técnica de reconexão de caminhos. O objetivo de inserir reconexão por caminhos a um algoritmo GRASP puro é armazenar soluções boas encontradas anteriormente, e utilizá-las como guias na busca por melhorias.

Reconexão por caminhos explora possíveis trajetórias conectando soluções de alta qualidade, obtidas por heurísticas como Busca Tabu e *Scatter Search* [9, 10, 11]. Seu primeiro uso integrado ao GRASP foi realizado por Laguna e Martí [16], e, atualmente, várias extensões, melhorias e aplicações de sucesso dessa técnica podem ser encontradas na literatura [5, 18].

De acordo com Rosseti [22], essa técnica é aplicada a um par de soluções $\{s_i, s_g\}$, no qual s_g é uma solução guia localmente ótima obtida após a fase de busca local, e s_i é uma solução selecionada aleatoriamente de um conjunto elite do GRASP, L, de tamanho $MaxElite$. Inicialmente, esse conjunto está vazio. Toda solução obtida pela busca local

é candidata a pertencer a L, desde que esta seja diferente de todas as outras soluções desse conjunto. Se $MaxElite$ é atingido e a candidata é melhor que a pior solução de L, a primeira substitui a segunda. Se o conjunto ainda não estiver completo, a candidata é inserida automaticamente.

O algoritmo inicia calculando a diferença simétrica $\triangle(s_i, s_g)$ entre s_i e s_g, cujo resultado é o conjunto de movimentos que s_i deve fazer para alcançar s_g. Partindo de s_i, realiza-se o melhor movimento ainda não executado até o momento em que s_g é atingida. Após esses movimentos, a melhor solução encontrada durante a trajetória é considerada candidata a pertencer ao conjunto elite, e a melhor solução global é atualizada.

Posteriormente, a partir do trabalho realizado por Rosseti [22], Barbalho *et al.* [4] propuseram dois métodos que utilizam mineração de dados no GRASP-RC. O conceito de Mineração de Dados (MD) consiste na extração automática de conhecimento a partir de grandes bases de dados [13]. O conhecimento extraído, expressado em forma de regras ou padrões, representa características importantes sobre uma base de dados. Nesse sentido, a mineração de dados provê um melhor entendimento sobre informações implícitas, auxiliando na compreensão de comportamentos e tomadas de decisão.

A partir dessa extração automática de conhecimento, Barbalho *et al.* [4] propuseram o método DM-GRASP-RC (*Data Mining* GRASP com reconexão por caminhos) a fim de, em certo momento, extrair padrões de um conjunto elite de soluções que conduzam e melhorem o processo de busca. Para um aprimoramento desse último método, Barbalho *et al.* [4] desenvolveram o método MDM-GRASP-RC (*Multi-Data Mining* GRASP com reconexão por caminhos) que realiza essa extração de padrões sempre que esse conjunto elite se tornar estável, e sempre que esse último for alterado e novamente se tornar estável.

No intuito de definir qual a melhor abordagem presente na literatura para resolver esse problema, experimentos computacionais foram realizados e foi comprovado, estatisticamente, que o três últimos algoritmos supracitados são superiores ao algoritmo guloso de Dahl e Johannessen [6] e, que o método MDM-GRASP-RC [4] possui o melhor desempenho em relação aos demais, em termos de tempo de processamento e qualidade de solução [4, 22].

O restante desta dissertação está organizada da seguinte maneira. No Capítulo 2 são apresentadas com mais detalhes as estratégias presentes na literatura para a resolução do 2-PNDP, inclusive o estado-da-arte MDM-GRASP-RC abordado em [4]. O Capítulo 3 apresenta dois novos métodos propostos neste trabalho para resolver o 2-PNDP, a saber: um método de pós-otimização e um método híbrido. O Capítulo 4 contém os experimentos

computacionais realizados, comparando e analisando o comportamento dessas duas novas abordagens propostas em relação à estratégia estado-da-arte. Finalmente, no Capítulo 5, é apresentada a conclusão deste trabalho e são propostos trabalhos futuros.

Capítulo 2

Heurísticas para o 2-PNDP

Neste capítulo, os algoritmos presentes na literatura que obtiveram mais sucesso na resolução do 2-PNDP são apresentados. São algoritmos que obtêm soluções aproximadas por meio de estratégias gulosas, técnicas de intensificação de soluções e mineração de dados.

2.1 Primeiro algoritmo guloso

Como mencionado no capítulo anterior, Dahl e Johannessen [6] propuseram uma formulação para o 2-PNDP definida como modelo 2-estrela, e uma estratégia gulosa baseada nesse modelo. Dessa forma, dado um par $(s,t) \in D$, essa estratégia mantém dois conjuntos de nós, S_s e S_t, com as seguintes características:

$$s \in S_s, s \notin S_t$$
$$t \in S_t, t \notin S_s$$

No pseudocódigo do Algoritmo 1, inicia-se com $S_s = \{s\}$, $S_t = \{t\}$ (linhas 2 e 3), c como sendo o custo da aresta $e \in E$ e $\overline{c}_e$ o custo reduzido da aresta $e \in E$. No início, $\overline{c} = c$ (linha 4) e são geradas duas 2-estrela (linha 7):

$$T_s = (S_s, V - S_s) \text{ e } T_t = (V - S_t, S_t)$$

Em seguida, é calculado o número m_e de 2-estrela que contém a aresta e (linhas 8 a 10) e computado $\epsilon^* = max\{\epsilon : \epsilon \cdot m_e \leq \overline{c}_e, \forall e \in E\}$ (linha 11). Após o cálculo de ϵ^*, o algoritmo atualiza o custo reduzido de cada aresta para $\overline{c}_e - \epsilon^* \cdot m_e$ (linhas 12 e 13). Se

o custo reduzido de uma aresta torna-se nulo (linha 14), essa aresta pertencerá à solução parcial gulosa (linha 15) e as suas extremidades farão parte dos conjuntos S_s e S_t, de acordo com suas regras de construção. Se as extremidades da aresta não pertencem ao conjunto S_s e ao conjunto S_t, um nó é atribuído ao conjunto S_s e o outro ao conjunto S_t (linhas 16 a 18). Se uma de suas extremidades pertence a S_t, a outra extremidade pertencerá a S_s (linhas 19 a 25). Se uma de duas extremidades pertence a S_s, a outra extremidade pertencerá a S_t (linhas 26 a 32).

Em seguida, todos os pares origem-destino que tenham um 2-caminho na solução gulosa são removidos do conjunto de demandas (linha 35) e, se esse conjunto estiver vazio, o algoritmo é finalizado (linha 6). Senão, uma nova iteração do laço das linhas 7 a 35 é realizada. Antes de retornar a solução gulosa construída (linha 38), o algoritmo remove todas as arestas que não afetam a viabilidade dessa solução (linha 37). O laço das linhas 6 a 36 é executado p vezes, no qual $p = |D|$. Os laços das linhas 8 a 10 e 12 a 34 são executados m vezes, nos quais $m = |E|$. Portanto, supondo-se que as demais instruções do algoritmo são $O(1)$, a complexidade desse algoritmo é $O(pm)$.

Nas próximas seções, são apresentadas outras estratégias gulosas presentes na literatura para a resolução do 2-PNDP.

2.2 GRASP com reconexão por caminhos para o 2-PNDP

No intuito de promover uma colaboração entre GRASP e RC, ambos os métodos foram instanciados para resolver o 2-PNDP. Nesta seção descreve-se como essas modificações foram implementadas.

2.2.1 Fase de construção

Nessa fase, o algoritmo de construção aleatório guloso computa um menor 2-caminho por vez [22]. O Algoritmo 2 mostra o pseudocódigo da fase de construção do GRASP para o 2-PNDP. Considere w_e como o conjunto dos pesos originais de cada aresta. Inicialmente, nas linhas 1 e 2, x é inicializado como conjunto vazio e w'_e recebe os pesos originais das arestas. A cada iteração, nas linhas 4 e 5, um par origem-destino $(a, b) \in D$ é escolhido aleatoriamente e o 2-caminho de menor custo para (a, b) é calculado utilizando os pesos w'_e. Em seguida, na linha 6, para cada aresta $(i, j) \in P$, seu peso w'_{ij} é definido como

Algoritmo 1: **Guloso_DJ [6]**

Seja (s, t) um par origem-destino de D;
$S_s \leftarrow \{s\}$;
$S_t \leftarrow \{t\}$;
$\overline{c} \leftarrow c$;
$S \leftarrow \emptyset$;
Enquanto $|D| \neq 0$ **faça**
  Gerar duas 2-estrela T_s e T_t;
  Para $\forall e \in E$ **faça**
    Calcular o número m_e de 2-estrela que contenham e;
  Fim-para
  $\epsilon^* = max\{\epsilon : \epsilon \cdot m_e \leq \overline{c}_e, \forall e \in E\}$;
  Para $\forall e \in E$ **faça**
    $\overline{c}_e \leftarrow \overline{c}_e - \epsilon^* \cdot m_e$;
    Se $\overline{c}_e = 0$ **então**
      $S \leftarrow S \cup \{e\}$;
      Sejam u e v as extremidades de e;
      Se $(u, v \notin S_s)$ e $(u, v \notin S_t)$ **então**
        $S_s \leftarrow S_s \cup \{u\}$; $S_t \leftarrow S_t \cup \{v\}$;
      senão
        Se $v \in S_t$ **então**
          $S_s \leftarrow S_s \cup \{u\}$;
        Fim-se
        Se $u \in S_t$ **então**
          $S_s \leftarrow S_s \cup \{v\}$;
        Fim-se
        Se $v \in S_s$ **então**
          $S_t \leftarrow S_t \cup \{u\}$;
        Fim-se
        Se $u \in S_s$ **então**
          $S_t \leftarrow S_t \cup \{v\}$;
        Fim-se
      Fim-se
    Fim-se
  Fim-para
  Remover de D todos os pares origem-destino que tenham um 2-caminho em S;
Fim-enquanto
Remover de S todas as arestas que não alterem sua viabilidade;
Retorne S;

zero. Nas linhas 7 e 8, esse par (a, b) é removido de D e o caminho P é anexado à solução S. Essa construção é interrompida quando D se torna vazio.

Algoritmo 2: pseudocódigo da fase de construção do GRASP para o 2-PNDP.

1: $S \leftarrow \emptyset$;
2: $w' \leftarrow w$;
3: **Enquanto** $D \neq \emptyset$ **faça**
4: Selecionar aleatoriamente um par origem-destino $(a, b) \in D$ ainda não atendido;
5: Computar o menor 2-Caminho P de a até b, usando os pesos w';
6: $w'_{ij} \leftarrow 0$ para todas arestas (i, j) em P;
7: $D \leftarrow D \setminus (\{a, b\})$;
8: $S \leftarrow S \cup P$;
9: **Fim-enquanto**
10: **Retorne** S;

Como mencionado anteriormente, a fase de construção do GRASP costuma não encontrar uma solução localmente ótima, necessitando a execução de uma busca local. A seguir, mais detalhes sobre essa segunda fase aplicada ao 2-PNDP são apresentados.

2.2.2 Busca local

Nessa busca local modificada cada solução S pode ser vista como uma coleção de $|D|$ 2-caminhos. Dada uma solução S, suas soluções vizinhas S' podem ser obtidas pela troca de qualquer 2-caminho de S por outro 2-caminho entre o mesmo par origem-destino. A fase de busca local tenta melhorar as soluções geradas de forma gulosa aleatória durante a fase de construção.

No pseudocódigo do Algoritmo 3 é explicado mais detalhadamente como ocorre esse processo. A solução vizinha S' e as arestas com pesos modificados são inicializados respectivamente nas linhas 1 e 2. A variável $semMudancas$ é inicializada na linha 3 e utilizada como uma *flag* que indica se um ótimo local foi atingido. Uma permutação circular contendo pares de demandas em D é criada aleatoriamente na linha 4. As instruções das linhas 5 a 16, são executadas até que todos $|D|$ 2-caminhos na solução atual tenham sido consecutivamente examinados e nenhum outro melhor 2-caminho tenha sido encontrado, indicando que a solução encontrada na busca local é um ótimo local.

Cada iteração inicia na linha 6 considerando o próximo par origem-destino (a, b), de acordo com a permutação circular criada na linha 4, e tentando melhorar seu 2-caminho. Dessa forma, os seguintes passos são executados: temporariamente redefinir para zero os pesos w' de todas as arestas utilizadas nos outros 2-caminhos (ver linha 7); computar o

menor 2-caminho entre a e b utilizando os pesos de arestas modificados w', dessa forma seus pesos não serão contabilizados mais de uma vez na solução, no caso em que arestas são reaproveitadas (ver linha 8); atualizar a melhor solução S' caso o peso do novo 2-caminho seja menor (ver linhas 9 e 10).

Uma vez que a iteração é finalizada, todos os pesos são redefinidos para seus valores originais w na linha 15. Se menos que $|D|$ 2-caminhos forem consecutivamente examinados sem melhoria na solução atual, então uma nova iteração é realizada. Caso contrário, a solução vizinha S' é retornada na linha 17.

Algoritmo 3: BuscaLocal2-caminho(x)

1: $S' \leftarrow S$
2: $w' \leftarrow w$
3: $semMudancas \leftarrow 0$
4: Criar aleatoriamente uma permutação circular de pares de demandas em D.
5: **Enquanto** $semMudancas < |D|$ **faça**
6: Selecionar o próximo par origem-destino (a, b) em D.
7: Temporariamente definir como 0 os pesos w' de todas as arestas que aparecem em 2-caminhos conectando os pares restantes em $D \setminus (a, b)$.
8: Computar o o menor 2-caminho entre (a, b) utilizando os pesos modificados w'.
9: **Se** o peso do novo 2-caminho for menor **então**
10: Atualizar a solução S' com o novo 2-caminho.
11: $semMudancas \leftarrow 0$
12: **senão**
13: $semMudancas \leftarrow semMudancas + 1$
14: **Fim-se**
15: Redefinir com o peso original w, todas as arestas definidas anteriormente com 0.
16: **Fim-enquanto**
17: **Retorne** S'

2.2.3 Reconexão por Caminhos

Nessa seção é mostrado como a RC foi integrada à metaheurística GRASP para o 2-PNDP. A reconexão por caminhos é aplicada na solução obtida pela busca local S_{bl}, e em uma solução S_{ce} selecionada aleatoriamente de L. Esse procedimento é executado duas vezes, uma utilizando S_{bl} como solução inicial e S_{ce} como guia (*Forward Relinking*), e outra definindo essa última como solução inicial e a primeira como guia (*Backward Relinking*) [18].

A solução localmente ótima obtida pela busca local e as melhores soluções encontradas ao longo de cada trajetória da reconexão são consideradas como candidatas a inserção no conjunto elite. Uma solução S_{bl} é inserida em L apenas se S_{bl} for diferente de todas as

outras soluções de L e seu custo for menor que o custo da pior solução em L.

O processo supracitado é apresentado com mais detalhes no pseudocódigo do Algoritmo 4. Na linha 1, o conjunto elite é inicializado como conjunto vazio e, na linha 2, o melhor custo encontrado até o momento, f^*, é definido como infinito. A partir da linha 3 até a linha 19, um bloco de instruções é executado $MaxIter$ vezes. Na linha 6, L é atualizado com a obtenção da nova solução S_{bl}, obtida na fase de construção na linha 4 e depois refinada pela busca local na linha 5. Na linha 7, se L possuir pelo menos duas soluções, então a reconexão por caminhos é executada das linhas 8 a 13. Em seguida, na linha 15, se uma solução com menor custo é encontrada pela reconexão de caminhos, a melhor solução é atualizada na linha 16. E, por fim, a melhor solução ao longo de todo o processo é retornada na linha 20.

Algoritmo 4: GRASP-RC($MaxIter$)

```
1: L ← ∅;
2: f* ← ∞;
3: Para 1 até MaxIter faça
4:    S ← Construcao2-Caminho();
5:    S_bl ← BuscaLocal2-Caminho(S);
6:    Atualizar o conjunto das soluções elite L com S_bl;
7:    Se |L| ≥ 2 então
8:       Seleciona aleatoriamente uma solução elite S_ce de L;
9:       S_1 ← ReconexaoPorCaminhos (S_bl, S_ce);
10:      Atualizar o conjunto das soluções elite L com S_1;
11:      S_2 ← ReconexaoPorCaminhos (S_ce, S_bl);
12:      Atualizar o conjunto das soluções elite L com S_2;
13:      S_bl ← argmin {f(S_bl), f(S_1), f(S_2) };
14:   Fim-se
15:   Se f(S_bl) < f* então
16:      S* ← S_bl;
17:      f* ← f(S_bl);
18:   Fim-se
19: Fim-para
20: Retorne S*;
```

2.3 Mineração de dados com GRASP-RC

Nesta seção, as estratégias que incorporam mineração de dados à heurística GRASP-RC são apresentadas.

2.3.1 Heurística DM-GRASP-RC

No GRASP original, as iterações são realizadas independentemente, e como consequência disso, o conhecimento adquirido em iterações anteriores não é utilizado nas subsequentes. A ideia básica da incorporação da mineração de dados é, em iterações anteriores, encontrar padrões em soluções de alta qualidade que conduzam e aprimorem o processo de busca.

Segundo Barbalho *et al.* [4], o método DM-GRASP-RC é composto por duas fases. A primeira é chamada de geração do conjunto elite, na qual consiste em executar j iterações do GRASP puro para obter um conjunto de diferentes soluções de boa qualidade. Em seguida, as b melhores soluções farão parte do conjunto elite do processo de mineração.

Após essa primeira fase, a técnica de mineração de dados é aplicada ao conjunto elite com o intuito de extrair padrões, ou seja, conjuntos de elementos que apareçam frequentemente nas soluções do seu conjunto elite. É importante ressaltar que um conjunto de itens frequentes com suporte $s\%$ representa um conjunto de elementos que ocorrem em pelo menos $s\%$ das soluções elite.

Posteriormente, a segunda fase do DM-GRASP-RC é realizada executando as demais j iterações, entretanto, com algumas diferenças em relação a fase anterior. Nessa nova fase, um padrão é selecionado a partir do conjunto de padrões minerados e, em seguida, uma fase de construção adaptada gerará uma solução completa a partir desse padrão, isto é, todas as soluções obtidas por essa construção conterão os elementos do padrão selecionado.

O DM-GRASP-RC desenvolvido por Barbalho *et al.* [4], que incorpora um processo de mineração de dados ao GRASP com reconexão por caminhos, mostra que não apenas o GRASP tradicional, mas como também o seu aprimoramento com reconexão por caminhos (um mecanismo de intensificação baseado em memória) , pode se beneficiar da utilização de padrões em soluções sub-ótimas para guiar a busca por melhorias no espaço de soluções.

O pseudocódigo do DM-GRASP-RC para o 2-PNDP é apresentado no Algoritmo 5. Na linha 3, o conjunto elite utilizado na mineração de dados M é inicializado como vazio. O laço das linhas 4 a 11 corresponde a fase de geração do conjunto elite da mineração, na qual o GRASP com reconexão por caminhos é executado por j iterações. O mesmo procedimento executado das linhas 4 a 14 do GRASP-RC é realizado na linha 5, seguida pela atualização do conjunto elite M, composto por b soluções, na linha 6. Uma solução é inserida no conjunto elite se já não estiver contida nele e for melhor que a pior solução desse

conjunto. Na linha 8, a melhor solução é atualizada, se a nova solução gerada possuir custo menor que a melhor encontrada até o momento. Na linha 12, o procedimento de mineração de dados extrai t padrões do conjunto elite, e tais padrões são inseridos em ordem decrescente de tamanho no conjunto de padrões. O laço das linhas 23 a 30, corresponde ao método apresentado nessa seção. Na linha 14, um padrão é selecionado do conjunto de padrões por meio de uma seleção *round-robin* [8].

Em seguida, o procedimento de construção adaptada é realizado na linha 15 e, na linha 16, a busca local é efetuada. Das linhas 18 a 23, o procedimento de reconexão por caminhos é executado. Se uma solução com menor custo é encontrada, a melhor solução atual é atualizada na linha 26. Após a execução de todas as iterações, a melhor solução é retornada na linha 30.

A extração de padrões a partir do conjunto elite, na qual é ativada na linha 23 do pseudocódigo mostrado no Algoritmo 5, corresponde a já bem conhecida tarefa de mineração de conjunto de itens frequentes (MCF). O problema MCF pode ser descrito da seguinte maneira [1]:

Seja $I = \{i_1, i_2, ..., i_n\}$ um conjunto de itens. Uma transação t é um subconjunto de I e um conjunto de dados B é um conjunto de transações. Um conjunto de itens frequentes F, com suporte s, é um subconjunto de I no qual ocorre em pelo menos $s\%$ das transações em B. O problema MCF consiste na extração de todos os conjuntos de itens frequentes de um conjunto de dados B com um mínimo de suporte especificado como parâmetro. Durante as últimas duas décadas, vários algoritmos têm sido propostos para minerar, de maneira eficiente, conjunto de itens frequentes [2, 12, 14, 17].

Na proposta dessa seção, os padrões a serem minerados são conjuntos de arestas que frequentemente aparecem em soluções sub-ótimas para o 2-PNDP. Essa é uma aplicação típica de mineração de conjuntos de itens frequentes, no qual o conjunto de itens é o conjunto de arestas potenciais (arestas com boas chances de pertencer a solução final) e cada transação de B representa uma solução sub-ótima do conjunto elite.

Um conjunto de itens frequentes é chamado de maximal se não há um super conjunto que também seja frequente. Com o intuito de evitar a mineração de um conjunto que seja subconjunto de outro, na proposta DM-GRASP-RC para o 2-PNDP extraiu-se apenas conjunto de itens frequentes maximais.

No Algoritmo 6, é apresentado o pseudocódigo da construção adaptada utilizado na segunda fase do DM-GRASP-RC. Ele é um pouco similar ao código descrito no Algoritmo

Algoritmo 5: DM-GRASP-RC(j, b, t)

1: $L \leftarrow \emptyset$;
2: $f^* \leftarrow \infty$;
3: $M \leftarrow \emptyset$;
4: **Para** $k = 1, ..., j$ **faça**
5: (Linhas 4 a 14 do Algoritmo 4)
6: atualizarElite(M, S_{bl}, b);
7: **Se** $f(S_{bl}) < f^*$ **então**
8: $S^* \leftarrow S_{bl}$;
9: $f^* \leftarrow f(S_{bl})$;
10: **Fim-se**
11: **Fim-para**
12: $conjunto\_padroes \leftarrow$ minerar (M, t)
13: **Para** $k = 1, ..., j$ **faça**
14: $padrao \leftarrow$ selecionarProximoMaiorPadrao($conjunto\_padroes$);
15: $S \leftarrow$ Construcao2-CaminhoAdaptado($padrao$);
16: $S_{bl} \leftarrow$ BuscaLocal2-Caminho(S);
17: Atualizar o conjunto das soluções elite L com S_{bl};
18: Seleciona aleatoriamente uma solução elite S_{ce} de L;
19: $S_1 \leftarrow$ ReconexaoPorCaminhos (S_{bl}, S_{ce});
20: Atualizar o conjunto das soluções elite L com S_1;
21: $S_2 \leftarrow$ ReconexaoPorCaminhos (S_{ce}, S_{bl});
22: Atualizar o conjunto das soluções elite L com S_2;
23: $S_{bl} \leftarrow$ argmin $\{f(S_{bl}), f(S_1), f(S_2)\}$;
24: **Se** $f(S_{bl}) < f^*$ **então**
25: $S^* \leftarrow S_{bl}$
26: $f^* \leftarrow f(S_{bl})$
27: **Fim-se**
28: **Fim-para**
29: **Retorne** S^*

2, com a diferença que, na linha 5, foi construído um 2-caminho entre o par (a, b) utilizando apenas as arestas pertencentes ao *padrao* ou com pesos modificados para zero. Se um 2-caminho não foi encontrado utilizando essas arestas, na linha 7, um 2-caminho é computado, iniciando-se a partir da solução parcial gerada até o momento na linha 5, e utilizando todas as outras arestas restantes de E.

Na proposta DM-GRASP-RC, o procedimento de mineração de dados é executado apenas uma vez, exatamente após metade das iterações. Apesar de obter resultados satisfatórios, Barbalho et al. [4] desenvolveram uma melhoria desse método, executando a mineração de dados mais de uma vez, assim que o conjunto elite se tornar estável e sempre que esse conjunto for alterado e novamente se tornar estável. Dessa forma, na seção seguinte, é apresentado o método MDM-GRASP-RC (*Multi Data-Mining* GRASP-RC) com desenvolvimento baseado na hipótese supracitada.

Algoritmo 6: Contrucao2-CaminhoAdaptada(*padrao*)

1: $x \leftarrow \emptyset$;
2: $w'_e \leftarrow w_e$;
3: **Enquanto** $D \neq \emptyset$ **faça**
4: Selecionar aleatoriamente um par origem-destino $(a, b) \in D$ ainda não atendido;
5: Computar o menor 2-Caminho P de a até b, usando arestas do *padrao* ou arestas com peso 0;
6: **Se** Se não foi possível encontrar um 2-Caminho P completo **então**
7: Computar o menor 2-Caminho P de a a b usando a solução parcial e as outras arestas ainda não utilizadas;
8: **Fim-se**
9: $w'_{ij} \leftarrow 0$ para todas arestas (i, j) em P;
10: $D \leftarrow D \setminus (\{a, b\})$;
11: $S \leftarrow S \cup P$;
12: **Fim-enquanto**
13: **Retorne** S ;

2.3.2 Heurística MDM-GRASP-RC

Um conjunto elite estável é um conjunto de soluções de alta qualidade que permanece inalterado por um determinado número de iterações, isto é, ao longo dessas iterações, não são encontradas soluções de boa qualidade que possam pertencer a esse conjunto.

A principal ideia do MDM-GRASP-RC é executar mineração de dados sempre que o conjunto elite se tornar estável, e também, sempre que esse conjunto for alterado e novamente ficar estável. A hipótese dessa seção sustenta a ideia que executar mineração de dados mais de uma vez explorará a evolução gradual do conjunto elite e permitirá a extração de padrões mais refinados.

Dessa forma, o pseudocódigo do MDM-GRASP-RC é apresentado no Algoritmo 7. A linha 6 corresponde à primeira fase de geração do conjunto elite, no qual iterações do GRASP com reconexão por caminhos são executadas até o momento que M esteja estável, ou o número máximo de iterações for alcançado. A seguir, das linhas 5 a 14, sempre que M estiver estável, o processo de mineração de dados é executado na linha 10. Na linha 12, o mesmo procedimento executado das linhas 14 a 27 do DM-GRASP-RC é realizado. Após todas as iterações, a melhor solução é retornada na linha 15.

No próximo capítulo é apresentada a proposta deste trabalho. Primeiramente é apresentada como foi realizada a análise das componentes de ambas estratégias vistas nessa seção e, em seguida, dois métodos são propostos baseados nessa análise, a saber: um método de pós-otimização para a solução obtida pela heurística original, e um método

híbrido que combina soluções encontradas pelo resolvedor de PLI com soluções elite da heurística original.

Algoritmo 7: MDM-GRASP-RC($maxIteracoes, d, t$)

1: $L \leftarrow \emptyset$;
2: $f^* \leftarrow \infty$;
3: $M \leftarrow \emptyset$;
4: $k \leftarrow 0$;
5: **Enquanto** $M\_nao\_estavel$ **e** $k < maxIteracoes$ **faça**
6: (Linhas 5 a 10 do Algoritmo 5);
7: **Fim-enquanto**
8: **Enquanto** $k < maxIteracoes$ **faça**
9: **Se** $M\_estavel$ **então**
10: $conjunto\_padroes \leftarrow$ minerar (M, t);
11: **Fim-se**
12: (Linhas 14 a 27 do Algoritmo 5);
13: $k \leftarrow k + 1$;
14: **Fim-enquanto**
15: **Retorne** S^*;

Capítulo 3

Estratégias propostas para o 2-PNDP

Esta dissertação possui como proposta utilizar PLI assistida por heurísticas baseadas em GRASP, reconexão por caminhos e mineração de dados. Para resolver o problema de síntese de redes a 2-caminhos, foram propostas duas estratégias que coletam informações no processamento da heurística original MDM-GRASP-RC [4], e utilizam essas informações para restringir o espaço de soluções do modelo caminho [6], acelerando o processamento do resolvedor de PLI, sem perda de qualidade de solução em relação às soluções alcançadas pela heurística original.

A primeira estratégia é um método de pós-otimização (MPO), no qual invoca-se o resolvedor de PLI após o término da execução do MDM-GRASP-RC [4]. Nesse caso, o modelo caminho, modificado pelas informações coletadas na execução dessa heurística, será resolvido a fim de se alcançar uma solução de melhor qualidade. A segunda estratégia é um método híbrido (HM75) no qual o resolvedor de PLI é chamado durante a execução da heurística original, a fim de melhorar a qualidade das soluções pertencentes aos conjuntos elite de mineração de dados e reconexão por caminhos.

3.1 Análises das execuções do MDM-GRASP-RC e GRASP-RC

Antes do desenvolvimento da proposta apresentada nesta seção, vale ressaltar que resolver o 2-PNDP por PLI pura com modelo caminho sem modificações revela-se impraticável, fato já esperado porque esse problema é NP-Difícil [6]. Apenas para realizar a sua relaxação linear, foram despendidas mais de três horas na tentativa de resolver uma instância de menor tamanho que contém 100 vértices e 1000 demandas.

Visto a ineficiência de se utilizar PLI pura para resolver essas instâncias, foi realizada uma análise acerca das execuções de ambas as estratégias GRASP-RC [22] e MDM-GRASP-RC [4]. Nessa investigação, observou-se uma maneira de reduzir o espaço de soluções do problema e fixar algumas arestas no modelo.

Durante as execuções das buscas locais, foi constatado que várias arestas (referidas neste trabalho como E^0) nunca são escolhidas para serem parte de uma solução ótima local. Para todas as instâncias abordadas em Barbalho *et al* [4], E^0 representa, no pior caso, 70% de E. Nesse sentido, removendo-se E^0 do grafo original, reduz-se o espaço de soluções do problema para $G(V, E \setminus E^0)$. Para uma ideia mais detalhada sobre o benefício da remoção do conjunto E^0, na Tabela 3.1 é apresentada uma instância de cada tamanho, mostrando a média das arestas removidas em dez execuções com sementes diferentes, e a porcentagem de E que essas arestas retiradas representam.

Instância	**Arestas removidas**	**Porcentagem de E**
a100-1	3485.9	70.42%
a200-1	15496.7	77.87%
a300-1	36172.9	80.65%
a400-1	66082.2	82.81%
a500-1	104238.4	83.56%

Tabela 3.1: Arestas removidas do problema original.

O pseudocódigo da geração do conjunto E^0 por meio de um GRASP básico é apresentado no Algoritmo 8, descrevendo o funcionamento básico da metaheurística GRASP, com exceção da linha 2, na qual o conjunto E^0 é inicializado como sendo o conjunto original de arestas E, e do bloco de instruções das linhas 6 a 10 após a busca local, no qual são removidas de E^0 todas as arestas presentes na solução S_{bl}, obtida por essa última busca local.

Após alguns experimentos utilizando o problema reduzido como entrada para o resolvedor de PLI, também observou-se que soluções obtidas por esse último e pela heurística original possuem conjuntos semelhantes de arestas mais frequentes. A frequência de uma aresta é determinada como o número de vezes que esta aresta foi utilizada para atender diferentes demandas. Essa informação serviu de guia para a hipótese de que ambas estratégias estariam realizando escolhas parecidas referentes ao conjunto de arestas mais promissoras, isto é, arestas com maiores probabilidades de estarem presentes na solução final.

Portanto, no intuito de impedir que o trabalho de escolha das melhores arestas seja

Algoritmo 8: **GRASP**(E^0)

```
1: S* ← ∅;
2: E^0 ← E;
3: Enquanto Critério de Parada faça
4:    S ← FaseDeConstrução();
5:    S_bl ← BuscaLocal(S);
6:    Para cada e ∈ S_bl faça
7:       Se e ∈ E^0 então
8:          E^0 ← E^0 \ {e};
9:       Fim-se
10:   Fim-para
11:   Se f(S_bl) < f(S*) então
12:      S* ← S_bl;
13:      f* ← f(S_bl);
14:   Fim-se
15: Fim-enquanto
16: Retorne S*;
```

realizado duas vezes, foram escolhidas, a partir da solução encontrada pelo MDM-GRASP-RC [4] (descritas no conjunto E^*), as arestas mais promissoras (referidas como o conjunto E^*_{final}), e foram fixadas suas respectivas variáveis no modelo de PLI com o valor igual a um, forçando o resolvedor a alcançar apenas soluções que contenham essas arestas e reduzindo, mais uma vez, o espaço de soluções. Na Tabela 3.2, são apresentados o número de arestas fixadas com o valor igual a um no modelo caminho, a porcentagem de E^* que o tamanho de E^*_{final} representa e o número de arestas livres, que representa o novo espaço de busca. No Capítulo 4 são apresentados os experimentos computacionais que justificam a escolha da porcentagem de E^*.

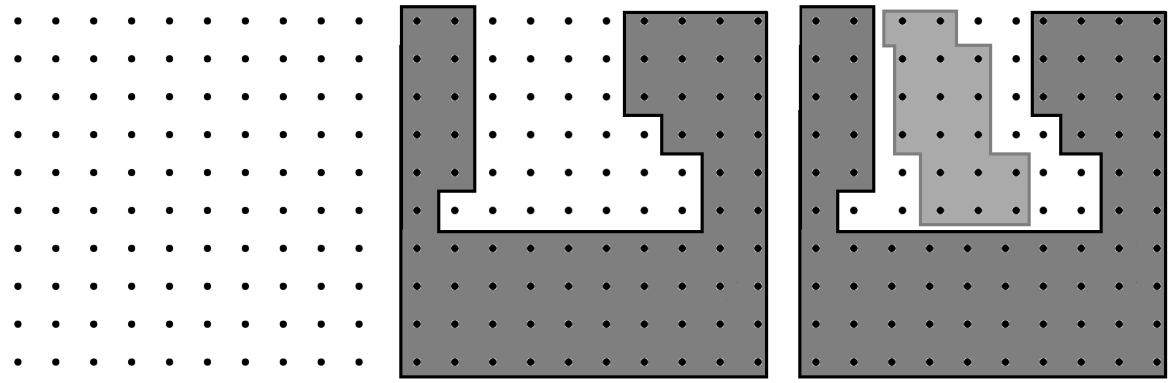
Instância	**Arestas fixadas**	**Porcentagem de E^* fixadas**	**Arestas livres**	**Porcentagem E livres**
a100-1	363.3	80%	1100.5	22.23%
a200-1	790.3	80%	3613.0	18.15%
a300-1	1303.7	80%	7373.4	16.44%
a400-1	1917.6	80%	11800.2	14.78%
a500-1	2559.7	80%	17951.9	14.39%

Tabela 3.2: Arestas fixadas e livres no modelo caminho.

Para um entendimento mais intuitivo sobre a redução do número de possíveis soluções para o problema, as Figuras 3.1(a) a 3.1(e) a seguir ilustram como ocorre o processo de redução. Para o exemplo apresentado a seguir, considere uma instância que possui 100 arestas, o tamanho de E^0 sendo 70% do tamanho de E e E^*_{final} como sendo 80% da solução obtida pelo MDM-GRASP-RC. Os pontos em preto na Figura 3.1(a) representam as arestas do grafo $G(V, E)$.

Gera-se o conjunto E^0 durante a execução do MDM-GRASP-RC, destacado na Figura 3.1(b) e após seu término, na Figura 3.1(c), a solução final é representada pelo conjunto E^*. O próximo passo realizado ilustrado na Figura 3.1(d) é descartar 20% das arestas menos promissoras de E^*, obtendo assim o conjunto E^*_{final} (destacado em cinza) referente a 80% das arestas restantes que obrigatoriamente pertencerão a solução final do MPO.

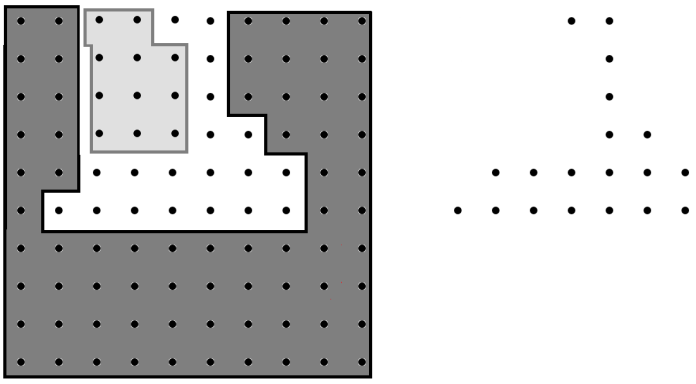
Concluindo a ilustração, a Figura 3.1(e) mostra o estado final do conjunto de arestas inicial E como sendo o novo espaço de busca.



(a) Representação do conjunto E.

(b) Seleção do conjunto E^0.

(c) Seleção da solução E^*.



(d) Seleção do conjunto E^*_{final}.

(e) Novo espaço de busca.

Figura 3.1: Reduções do espaço de busca.

No contexto de programação linear inteira, as eliminações de arestas ilustradas nas

figuras anteriores podem ser interpretadas como reduções do poliedro que representa o 2-PNDP. Porém, não há garantia de otimalidade da solução. Dessa forma, uma nova versão do modelo caminho para o 2-PNDP pode ser descrita como segue:

$$min \sum_{e \in E} c_e x_e \tag{3.1}$$

sujeito a:

$$\sum_{p \in P_d} z_p = 1, \qquad \forall d \in D \tag{3.2}$$

$$z_P \leq x_e, \qquad \forall d \in D, P \in P_d, e \in P \tag{3.3}$$

$$z_P \geq 0, \qquad \forall d \in D, P \in P_d \tag{3.4}$$

$$\sum_{\forall e \in E^*_{final}} x_e = |E^*_{final}|, \tag{3.5}$$

$$\sum_{\forall e \in E^0} x_e = 0, \tag{3.6}$$

$$x_e, z_P \in \{0, 1\}, \qquad \forall d \in D, P \in P_d, e \in P \tag{3.7}$$

Para esta nova formulação, dois conjuntos de restrições foram adicionados. A primeira restrição (3.5) exige que todas as arestas do conjunto E^*_{final} pertençam a todas as soluções encontradas pelo resolvedor ao longo do processo de otimização, e a segunda, restrição (3.6), impede que qualquer aresta contida em E^0 esteja presente em alguma solução obtida pelo resolvedor.

A partir desses dois novos conjuntos de restrições, reduziu-se consideravelmente o tempo de processamento do modelo caminho, sem perda de qualidade de solução em relação à solução alcançada pelo MDM-GRASP-RC. No entanto, como o espaço de busca não se encontra completo, não se pode garantir a otimalidade do modelo.

A análise realizada a partir do MDM-GRASP-RC é um processo que pode ser aplicado a outras estratégias que resolvam problemas de otimização combinatória. Para tal análise, é importante que a estratégia utilize heurísticas de busca para a formação do conjunto E^0, e que possua um bom critério de seleção de arestas promissoras para a formação do conjunto E^*_{final}.

Por meio da análise de execução do MDM-GRASP-RC, foi proposto um método de pós-otimização (MPO) que resolverá o 2-PNDP com seu espaço de busca reduzido, base-

ado em informações obtidas durante a execução da heurística original. Após a execução do MDM-GRASP-RC e construção dos conjuntos E^* e E^0, a pós-otimização é realizada. O MPO consiste em: (a) selecionar as arestas mais promissoras E^*_{final} da solução obtida pela heurística original; (b) fixar suas respectivas variáveis com o valor um no modelo; (c) remover do modelo todas as variáveis que representam as arestas pertencentes ao conjunto E^0; (d) executar o resolvedor de PLI recebendo, como parâmetro de entrada, a nova versão do modelo caminho.

O pseudocódigo do Algoritmo 9 mostra com mais detalhes este processo de pós-otimização proposto. Além de E^0 e E^*, são passados, como parâmetros de entrada, o modelo caminho e a variável $numFix$, definindo a porcentagem da solução final do MDM-GRASP-RC que pertencerão à solução final da pós-otimização, determinando, assim, um limite inferior de similaridade entre as soluções obtidas pelo MDM-GRASP-RC e pelo método de pós-otimização. Na linha 1, é chamada uma função na qual ordenará todas as arestas pertencentes a E^*, de acordo com os critérios de *incidencia*, correspondente ao custo da aresta, e *num_vezes*, o número de vezes que uma aresta é utilizada no atendimento às demandas.

Uma vez que as melhores arestas se encontram ordenadas na lista $arestas\_ordenadas$, na linha 2, a variável f é inicializada com o número de arestas do conjunto E^* que serão utilizadas. Das linhas 4 a 7, as f primeiras arestas da lista $arestas\_ordenadas$ são inseridas no conjunto E^*_{final}. Na linha 8, a modificação do modelo caminho utilizando os conjuntos E^0 e E^*_{final} é realizada. Na linha 9 o resolvedor é invocado recebendo, como parâmetro de entrada, o modelo modificado. E, por fim, a solução obtida após a execução do resolver é retornada na linha 10.

Algoritmo 9: **MPO**$(E^*, E^0, numFix, modelo)$

1: ordenar_decrescente$(E^*, arestas_ordenadas, num_vezes, incidencia)$;
2: $f \leftarrow |arestas\_ordenadas| \times num\_fix$;
3: $i \leftarrow 0$;
4: **Enquanto** $i < f$ **faça**
5: $\quad E^*_{final} \leftarrow E^*_{final} \cup arestas\_ordenadas(i)$;
6: $\quad i \leftarrow i + 1$;
7: **Fim-enquanto**
8: fixar$(modelo, E^0, E^*_{final})$;
9: $S_{final} \leftarrow$ resolvedor$(modelo)$;
10: **Retorne** S_{final};

3.2 Método híbrido

Além do método de pós-otimização descrito anteriormente, neste trabalho também é proposta uma heurística híbrida que utiliza o resolvedor de PLI durante o processamento da heurística MDM-GRASP-RC. Esta proposta tem o objetivo de coletar informações na execução do MDM-GRASP-RC, e melhorar a qualidade de solução dos conjuntos elite da heurística original.

Nesta nova estratégia (HM75) o resolvedor recebe informações intermediárias em relação às informações recebidas pelo MPO. No método HM75, E^* passa a ser a melhor solução encontrada até o momento pelo MDM-GRASP-RC, e E^0 o conjunto das arestas não utilizadas em nenhuma busca local até o momento da chamada do resolvedor. Após solucionar o problema com espaço de busca reduzido, o resolvedor retorna um conjunto de soluções de boa qualidade e tenta inseri-las nos conjuntos elite da mineração de dados M e da reconexão por caminhos L e, em seguida, o processamento do MDM-GRASP-RC continua sem modificações em relação à sua versão original. O pseudocódigo do Algoritmo 10 apresenta com mais detalhes a proposta deste novo método híbrido.

As primeiras iterações do HM75 assemelham-se com as iterações do MDM-GRASP-RC, porém, com a construção do conjunto E^0 (ver Algoritmo 8) que ocorre entre as linhas 8 e 12. Na linha 13, ao se atingir a iteração correspondente ao *momento_de_chamada* pré-fixado, são executados os seguintes passos: (a) é realizado o mesmo processo de fixação e remoção de arestas executado pelo MPO (ver linha 14), contudo, no HM75, o resolvedor retorna um conjunto *vec_sols* contendo todas as soluções inteiras obtidas (ver linha 15); (b) a melhor solução global é atualizada a partir das soluções pertencentes a *vec_sols* (ver linhas 17 a 23); (c) os conjuntos elite M e L são atualizados a partir das soluções pertencentes a *vec_sols*; e (d) a partir da linha 27, o método híbrido continua o procedimento original do MDM-GRASP-RC.

No capítulo seguinte são apresentados resultados computacionais obtidos pela estratégia de Barbalho *et al.* [4] e pelas estratégias MPO e HM75 propostas neste capítulo.

Algoritmo 10: **HM75**$(maxIteracoes, d, t, E^*, E^0, numFix)$ Fase 2

1: **Enquanto** $k < maxIteracoes$ **faça**
2: **Se** M *estavel* **então**
3: $conjunto\_padroes \leftarrow$ executar_mineracao (M, t);
4: **Fim-se**
5: $padrao \leftarrow$ selecionarProximoMaiorPadrao($conjunto\_padroes$) ;
6: $S \leftarrow$ Construcao2-CaminhoAdaptado($padrao$);
7: $S_{bl} \leftarrow$ BuscaLocal2-Caminho(S);
8: **Para cada** $e \in S_{bl}$ **faça**
9: **Se** $e \in E^0$ **então**
10: $E^0 \leftarrow E^0 \setminus \{e\}$;
11: **Fim-se**
12: **Fim-para**
13: **Se** $momento\_de\_chamada$ **então**
14: (Linhas 1 a 8 do Algoritmo 9);
15: $vec\_sols() \leftarrow$ resolvedor($modelo$);
16: $i \leftarrow 0$;
17: **Enquanto** $i < |vec\_sols|$ **faça**
18: **Se** $f(vec\_sols(i)) < f^*$ **então**
19: $x^* \leftarrow vec\_sols(i)$;
20: $f^* \leftarrow f(vec\_sols(i))$;
21: **Fim-se**
22: $i \leftarrow i + 1$;
23: **Fim-enquanto**
24: Atualizar o conjunto das soluções elite L com $vec\_sols()$;
25: atualizarElite$(M, vec_sols(), b)$;
26: **Fim-se**
27: (Linhas 17 a 27 do Algoritmo 5);
28: $k \leftarrow k + 1$
29: **Fim-enquanto**
30: **Retorne** S^*

Capítulo 4

Resultados computacionais

Neste capítulo são comparados os resultados computacionais obtidos pela heurística MDM-GRASP-RC [4] com as propostas apresentadas no Capítulo 3, denominadas MPO e HM75, a fim de avaliar a utilização de programação linear inteira juntamente com a heurística proposta por Barbalho *et al.* [4].

Este capítulo está organizado da seguinte maneira. Na Seção 4.1, são apresentados o ambiente de execução e as instâncias utilizadas para os experimentos computacionais. Os resultados das estratégias MPO e HM75 são comparados com o MDM-GRASP-RC na Seção 4.2. Testes estatísticos são empregados a fim de verificar a significância estatística desses resultados. Por fim, nas Seções 4.2.2 e 4.3.2, são realizadas análises de comportamento para as três estratégias a fim de comprovar o benefício da utilização de PLI incorporada à heurística de Barbalho *et al.* [4].

4.1 Ambiente de execução e instâncias utilizadas

Todos os métodos abordados nesta seção foram implementados na linguagem de programação C e, uma vez que os autores do MDM-GRASP-RC disponibilizaram o código-fonte do seu algoritmo, utilizou-se o mesmo compilador gcc versão 4.8.3 para as três estratégias. Os experimentos computacionais foram realizados em um computador pessoal Intel®Core™ i5 CPU 650 @ 3.20 GHz com 4Gb RAM, utilizando o sistema operacional Linux. Para a execução do modelo de PLI, utilizou-se o IBM ILOG CPLEX Optimizer [15] com suas heurísticas e cortes para aprimoramento de soluções desativados, e a capacidade de paralelização do processador não foi aproveitada.

Para a execução dos experimentos foram utilizadas as mesmas instâncias apresentadas

em Barbalho *et al.* [4]. Tratam-se de 25 instâncias semelhantes às instâncias geradas em [20], sendo grafos completos com $|V| \in \{100, 200, 300, 400, 500\}$, com custos das arestas gerados aleatoriamente a partir de uma distribuição uniforme no intervalo $[1, 10]$, e 10 $\times|V|$ pares origem-destino escolhidos de maneira aleatória.

4.2 Resultados para a estratégia de pós-otimização (MPO)

Como mencionado anteriormente no Capítulo 3 a porcentagem de arestas ($numFix$) escolhidas, a partir da solução original obtida pela abordagem MDM-GRASP-RC, é um parâmetro que indica um nível mínimo de similaridade entre a solução encontrada pelo resolvedor e a solução obtida pela heurística original.

Dessa forma, a fim de encontrar a configuração com melhor custo benefício, foram escolhidas cinco instâncias e, para cada uma, o parâmetro $numFix$ foi fixado em 60%, 70%, 75%, 80% e 90%. Para cada instância, dez execuções foram realizadas com limite de tempo de 3600 segundos e os resultados são apresentados na Tabela 4.1. A primeira coluna corresponde ao identificador de instância $ax - y$ utilizado por Barbalho *et al.* [4], na qual $x = |V|$ e y é a semente usada como parâmetro para gerar uma instância aleatória. A segunda coluna corresponde a melhor solução conhecida (MSC) na literatura [4]. As colunas seguintes representam o melhor custo, a média de custo e a média de tempo extra utilizada pelo resolvedor após a execução do MDM-GRASP-RC, para cada valor de $numFix$.

		60%			70%			75%			80%			90%		
Instância	MSC	Melhor Solução	Média Solução	Tempo Extra	Melhor Solução	Média Solução	Tempo Extra	Melhor Solução	Média Solução	Tempo Extra	Melhor Solução	Média Solução	Tempo Extra	Melhor Solução	Média Solução	Tempo Extra
a100-1	676	665	671.90	3600.00	664	673.20	1209.60	671	676.90	50.40	675	678.80	0.90	675	679.70	0.44
a200-1	1374	1373	1381.30	3600.00	1367	1372.10	236.10	1367	1373.10	17.90	1367	1373.30	46.90	1374	1381.10	1.56
a300-1	2082	2068	2093.33	3600.00	2065	2081.10	2080.80	2072	2082.10	1547.60	2069	2086.10	81.40	2080	2093.10	3.57
a400-1	2779	2782	2787.67	3600.00	2792	2801.60	1713.20	2773	2783.00	1413.50	2774	2782.10	27.10	2777	2785.00	6.23
a500-1	3554	3773	3566.44	3600.00	3567	3680.30	3600.00	3541	3555.20	2003.70	3548	3561.70	22.00	3552	3565.50	10.02
Média				3600.0			1768.62			1006.62			35.66			3.57

Tabela 4.1: Porcentagem de arestas fixadas.

Na Tabela 4.1, à medida que o nível de fixação aumenta, o custo de solução piora e o tempo de processamento do resolvedor diminui em relação a níveis menores de fixação. Utilizando a fixação de 80%, o MPO já alcança valores de soluções melhores que as melhores soluções conhecidas para essas cinco instâncias. Esses últimos resultados requerem apenas um pequeno tempo adicional de execução em relação às outras configurações (com média de 35.66 segundos).

Para verificar que os custos das soluções alcançadas pelos algoritmos comparados não seguem uma distribuição normal, foi usado o teste de normalidade *Shapiro-Wilk* [23]. Para realizar esse teste, foram feitas duas hipóteses:

- Hipótese nula (H_0): Os custos das soluções obtidas seguem uma distribuição normal; e
- Hipótese alternativa (H_1): Os custos das soluções obtidas não seguem uma distribuição normal;

Usando o pacote R [25] com o nível de significância do teste α igual a 0.05, foram encontrados os valores de p-valor igual a 0.000274 para as soluções alcançadas pelo algoritmo MDM-GRASP-RC, e p-valor igual a 0.0002776 para as soluções obtidas pela estratégia MPO. Como os p-valores calculados são menores que α, pode-se concluir, com 95% de confiança, que os custos das soluções alcançadas pelas duas abordagens não seguem a distribuição normal.

Para comprovar a melhoria de custo a partir da variação da fixação, foi realizado o teste estatístico não paramétrico de *Friedman* [24]. Esse teste é uma alternativa ao teste estatístico t, quando a amostra analisada não segue a distribuição normal. Para utilizar o teste de *Friedman*, foram desconsideradas as variações de num_fix 60% e 70% em que o tempo limite de 3600 segundos foi excedido em pelo menos uma das cinco instâncias. Geralmente, o teste de *Friedman* [24] é utilizado para avaliar algoritmos que possuem componentes aleatórias, e identificar se a diferença entre suas médias foi devido à superioridade de algum desses algoritmos ou apenas devido à aleatoriedade presente nos métodos. Para a realização desse teste, foram considerados p-valor igual a 0.05 e duas hipóteses:

- Hipótese nula (H_0): Não há diferença entre as médias encontradas pelas configurações $numFix$; e

- Hipótese alternativa (H_1): Há diferença entre as médias encontradas pelas configurações $numFix$;

Nesse sentido, H_0 será rejeitada com 95% de certeza se, para cada instância, o teste de *Friedman* [24] retornar um valor menor ou igual ao p-valor definido. Caso H_0 seja rejeitada, a hipótese alternativa H_1 é considerada.

Na Tabela 4.2, é apresentada para cada par de num_fix e para cada instância utilizada, o valor igual a um, se o respectivo num_fix obteve melhor média de solução, e zero caso contrário. Dentro dos parênteses, o valor igual a um indica que essa melhor média foi obtida com p-valor menor que 0.05, significando que a probabilidade da diferença de desempenho entre as variações comparadas ser devido à aleatoriedade é menor que 5%, e zero caso contrário.

Para essa análise estatística, realizou-se o teste de *Friedman* [24] a partir do num_fix 75%, pois as configurações 60% e 70% excedem o tempo limite para pelo menos uma instância (ver Tabela 4.1). A partir da comparação entre 75% e 80%, percebe-se que: (a) o num_fix 75% obteve melhor custo médio para quatro instâncias, porém, sem significância estatística para todos esses casos; e, (b) que a configuração de 80% foi superior à 75% para a instância a400-1, com significância estática e, como a configuração de 75% despende um tempo de processamento muito maior que a primeira, num_fix foi definido para 90% e um novo teste foi realizado.

Nesse novo teste comparando 80% com 90%, a utilização do menor num_fix resulta em melhor média para todas as cinco instâncias, com significância estatística para quatro. Apesar do tempo menor despendido pela configuração de 90% (ver Tabela 4.1), nota-se que a diferença de tempo despendido por essas duas configurações possui impacto inferior em relação à melhoria de custo realizada com o num_fix igual a 80%. Portanto, definiu-se 80% como a configuração padrão de fixação no MPO.

num_fix	Instâncias				
	a100-1	a200-1	a300-1	a400-1	a500-1
75%	1(0)	1(0)	1(0)	0(0)	1(0)
80%	0(0)	0(0)	0(0)	1(1)	0(0)
80%	1(0)	1(1)	1(1)	1(0)	1(1)
90%	0(0)	0(0)	0(0)	0(0)	0(0)

Tabela 4.2: Análise de significância estática para validação do num_fix.

Após o experimento para escolher o grau de fixação, a abordagem proposta foi testada para as instâncias restantes. Como apresentado no Capítulo 3, o método de pós-otimização

é realizado ao final da heurística MDM-GRASP-RC e, portanto, seu tempo computacional começa a ser medido a partir do tempo despendido por essa última abordagem.

Dessa forma, para uma comparação justa, o tempo extra despendido em cada uma das execuções do MPO foi fornecido ao MDM-GRASP-RC, modificando, assim, seu critério de parada, isto é, a abordagem original agora interrompe a sua execução quando é alcançado o tempo utilizado pela heurística proposta MPO.

Os resultados dessa comparação são apresentados na Tabela 4.3. Cada estratégia foi executada dez vezes para cada instância, com sementes diferentes. A Tabela 4.3 apresenta o custo da melhor solução obtida, o custo médio referente às dez execuções, o tempo computacional médio utilizado nos dois métodos e os desvios percentuais de ambas estratégias em relação aos critérios melhor solução e média. Na comparação entre as duas abordagens, os valores em negrito representam os melhores resultados alcançados. Na Tabela 4.3, o desvio percentual de um critério ser igual a zero, com precisão de três casas decimais, indica que a estratégia analisada obteve o melhor resultado da comparação para esse critério.

Para o cálculo desse desvio percentual, considera-se a Equação 4.1.

$$DP_{criterio} = \frac{resultado(criterio) - melhor\_resultado(criterio)}{melhor\_resultado(criterio)} \tag{4.1}$$

Observaessendo a Tabela 4.3, comprova-se que realmente o MPO possui um desempenho melhor ou igual ao MDM-GRASP-RC para a maioria das instâncias, usando-se a mesma quantidade de tempo de execução em ambas as abordagens.

A redução no custo das soluções obtidas pelo MPO deve-se ao fato deste método sempre convergir para um ótimo global do problema modificado (problema original com a remoção das arestas em E^0 e fixação das arestas em E^*), em função das informações adquiridas. O mesmo muitas vezes não acontece com o MDM-GRASP-RC, que em determinado momento de sua execução, possui essas mesmas informações das execuções já realizadas, e apenas atinge um ótimo local.

4.2.1 Significância estatística

A fim de verificar se as diferenças dos valores médios de custo, mostrados na Tabela 4.3, obtidos pelas estratégias avaliadas são estatisticamente significantes, foi realizado o mesmo teste estatístico não paramétrico de *Friedman* [24] apresentado anteriormente

Instância	Melhor Solução	Melhor Média	Tempo Médio	MDM-GRASP-RC Melhor Solução	MDM-GRASP-RC Média Solução	MDM-GRASP-RC DP Melhor	MDM-GRASP-RC DP Média	MPO Melhor Solução	MPO Média Solução	MPO DP Melhor	MPO DP Média
a100-10	655	665.1	9.37	658	666.8	0.004	0.002	**655**	**665.1**	0.000	0.000
a100-100	661	667.2	10.27	662	669.0	0.001	0.002	**661**	**667.2**	0.000	0.000
a100-1000	641	646.3	9.54	642	648.4	0.001	0.003	**641**	**646.3**	0.000	0.000
a100-10000	658	663.5	9.71	660	665.9	0.003	0.003	**658**	**663.5**	0.000	0.000
a200-10	1362	1363.0	55.41	**1362**	1371.0	0.000	0.005	**1362**	**1363.0**	0.000	0.000
a200-100	1339	1348.8	59.99	1344	1352.0	0.003	0.0002	**1339**	**1348.8**	0.000	0.000
a200-1000	1349	1356.2	67.81	1354	1362.5	0.003	0.004	**1349**	**1356.2**	0.000	0.000
a200-10000	1357	1365.6	50.89	1360	1370.7	0.002	0.003	**1357**	**1365.6**	0.000	0.000
a300-10	2113	2119.9	134.77	2119	2124.9	0.002	0.002	**2113**	**2119.9**	0.000	0.000
a300-100	2062	2067.0	124.37	2065	2072.8	0.001	0.002	**2062**	**2067.0**	0.000	0.000
a300-1000	2067	2082.1	276.23	**2067**	**2082.1**	0.000	0.000	2073	2082.9	0.002	0.000
a300-10000	2053	2064.9	133.02	2054	2069.8	0.000	0.002	**2053**	**2064.9**	0.000	0.000
a400-10	2826	2838.7	234.13	2832	**2838.7**	0.002	0.000	**2826**	2933.8	0.000	0.030
a400-100	2791	2797.1	232.24	2796	2801.4	0.001	0.001	**2791**	**2797.1**	0.000	0.000
a400-1000	2789	2801.3	241.43	2792	2804.4	0.001	0.001	**2789**	**2801.3**	0.000	0.000
a400-10000	2821	2839.3	243.51	2827	2845.1	0.002	0.002	**2821**	**2839.3**	0.000	0.000
a500-10	3553	3562.7	395.94	3556	3565.7	0.000	0.000	**3553**	**3562.7**	0.000	0.000
a500-100	3556	3569.4	424,58	3564	3572.8	0.000	0.000	**3556**	**3569.4**	0.000	0.000
a500-1000	3527	3546.8	420.90	3531	3551.5	0.001	0.001	**3527**	**3546.8**	0.000	0.000
a500-10000	3550	3580.2	414.83	3556	3584.7	0.001	0.001	**3550**	**3580.2**	0.000	0.000

Tabela 4.3: Resultados computacionais comparando MDM-GRASP-RC e MPO utilizando o mesmo tempo de execução.

nesta seção.

Estratégia	Grupo de instâncias				
	a100	a200	a300	a400	a500
MDM-GRASP-RC	0 (0)	0 (0)	1 (0)	1 (1)	0 (0)
MPO	4(4)	4(3)	3(3)	3(3)	4(4)

Tabela 4.4: Análise de significância estatística

A Tabela 4.4 apresenta, para cada par de estratégia e para cada grupo de instâncias (instâncias com o mesmo número de nós), o número de instâncias com melhores médias encontradas por cada estratégia e, entre parênteses, o número de vezes em que p-valor foi menor que 0.05, significando que a probabilidade da diferença de desempenho entre os algoritmos comparados ser devido à aleatoriedade é menor que 5%. Percebe-se que quase todos os resultados de desempenho melhor possuem significância estatística, com exceção do grupo a200, no qual um resultado de desempenho melhor para o MPO (em relação a estratégia original) não possui significância estatística. Já para uma instância de 300 nós, apesar do MDM-GRASP-RC ser melhor que a MPO, esse resultado não

possui significância estatística. O único caso em que o MDM-GRASP-RC foi melhor com significância estatística foi para uma instância do grupo a400.

4.2.2 Análise do comportamento das estratégias

Nesta seção, é apresentada uma análise adicional aos experimentos computacionais a fim de ilustrar o comportamento das estratégias. Realizou-se uma comparação entre os dois métodos MPO e MDM-GRASP-RC baseada nos gráficos *Time-to-target plots* (`TTTplots`) [3], usado para analisar o comportamento de algoritmos com componentes aleatórias. Basicamente, esses gráficos mostram a probabilidade acumulada de um algoritmo encontrar uma solução melhor ou igual a uma solução alvo prefixada, em um certo tempo de execução.

O gráfico mostrado na Figura 4.1 foi gerado pelas execuções do MDM-GRASP-RC e MPO a partir da instância a100-1, utilizando um alvo de valor igual a 675. Para esse alvo, o MPO apresenta um desempenho superior ao MDM-GRASP-RC. Pode-se observar na Figura 4.1 que o MPO atinge o alvo em 9.73 segundos com cerca de 90% de probabilidade, enquanto o MDM-GRASP-RC possui probabilidade de apenas 60% aproximadamente para esse mesmo tempo de execução.

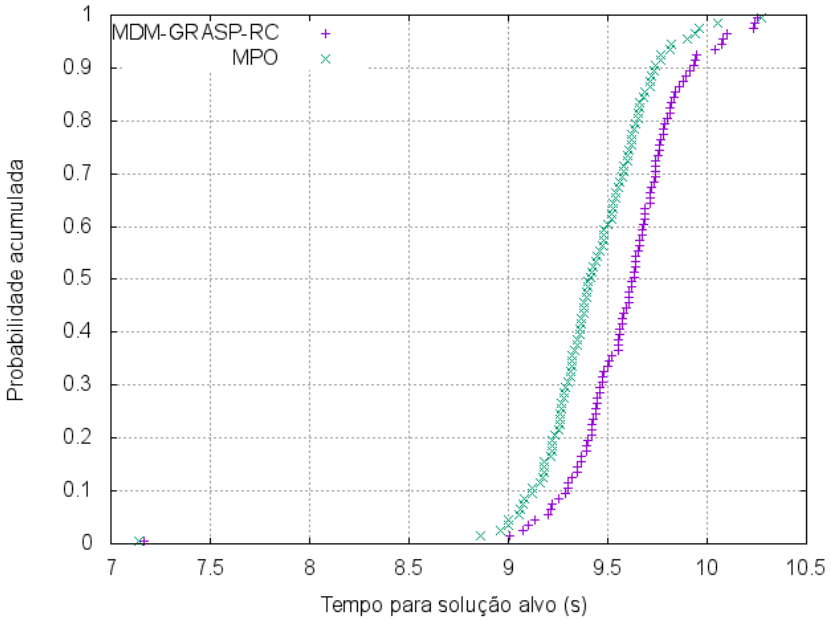


Figura 4.1: `TTTplots` para as estratégias MDM-GRASP-RC e MPO.

4.3 Resultados da estratégia híbrida (HM75)

Apesar da proposta de pós-otimização ter obtido resultados superiores aos obtidos pelo MDM-GRASP-RC, optou-se por outra maneira de combinar PLI com a proposta de Barbalho *et al.* [4]. A partir da hipótese de que se poderia melhorar a qualidade das soluções pertencentes aos conjuntos elite M e L utilizando as soluções obtidas pelo resolvedor de PLI. Esse resolvedor foi chamado uma vez em algum momento na heurística original, em um certo *momento_de_chamada* e as soluções obtidas após as chamadas do resolvedor foram inseridas nesses conjuntos elite.

No início dos experimentos computacionais, tentou-se definir o *momento_de_chamada* como sendo uma iteração que ocorresse já nas primeiras iterações do método HM75. Porém, o resolvedor não foi capaz de obter soluções melhores que a melhor solução encontrada até o momento S^*. O baixo desempenho do resolvedor chamado nessas primeiras iterações deve-se a má qualidade das informações coletadas no início do HM75 para os conjuntos E^0 e E^*, uma vez que a qualidade desses conjuntos são diretamente proporcionais ao número de iterações do HM75.

No caso do conjunto E^0, quanto maior o número de iterações realizadas pelo método HM75, maior o número de buscas locais e, consequentemente, menor o tamanho do conjunto E^0. Ao decorrer do tempo de execução do HM75, apenas arestas consideradas não promissoras (isto é, provavelmente não pertencerão à solução final) permanecem em E^0.

No caso do conjunto E^*, que contém as arestas da melhor solução alcançada até o *momento_de_chamada*, a qualidade desse conjunto aumenta com o avanço das iterações e com a convergência para soluções de melhor qualidade. Portanto, observou-se que geralmente após um certo número pequeno de iterações, a qualidade das soluções obtidas ainda não era promissora o suficiente para determinar quais arestas do conjunto E^* devem pertencer ao conjunto E^*_{final} (conjunto de arestas que farão parte da solução final do HM75).

À medida que o *momento_de_chamada* se aproxima das iterações finais, a solução final desse método assemelha-se à solução obtida pelo MPO, no qual o resolvedor é invocado logo após o término da heurística original. Essa nova proposta tem por objetivo encontrar soluções melhores e possivelmente diferentes das soluções obtidas pelo MPO. Dessa forma, a fim de encontrar um momento em que a qualidade dos conjuntos E^0 e E^* seja razoável sem comprometer a qualidade das soluções alcançadas, foram realizados experimentos computacionais variando o *momento_de_chamada* a partir das iterações finais do HM75

para três de cada grupo de instâncias (Tabela 4.5).

Iniciou-se invocando o resolvedor no momento que corresponde a 95% das iterações iterações do HM75. Por exemplo, para essa configuração de 95%, se o valor de $maxIteracoes$ fosse igual a 1000, o resolvedor seria chamado na iteração 950. Após esse primeiro experimento, variou-se o momento de chamada também para 90%, 85%, 80%, 75% e 70%, e uma comparação entre essas configurações foi realizada. Na Tabela 4.5, para essa comparação, são apresentados os melhores custos de soluções e as médias de custo, e os valores destacados em negrito representam os melhores resultados. Nesse experimento computacional, as médias de tempo não foram consideradas.

Instância	95%		90%		85%		80%		75%		70%	
	Média Solução	Melhor Solução	Média Solução	Melhor Solução	Média Solução	Melhor Solução	Média Solução	Melhor Solução	Média Solução	Melhor Solução	Média Solução	Melhor Solução
a100-1	679.33	675	678.67	675	678.33	672	678.25	671	**669**	**677.10**	671	678.25
a100-10	668.00	655	664.50	655	663.50	655	668.00	655	**655**	**663.10**	**655**	668.00
a100-100	666.80	661	666.70	659	665.40	659	665.80	659	**659**	666.20	**659**	**665.80**
a200-1	1370.33	1368	1370.67	1366	1370.64	1366	1369.91	1365	**1365**	1370.00	**1365**	**1369.91**
a200-10	1362.30	1351	1361.30	1352	1361.20	1350	1357.00	1351	**1348**	1359.70	1351	**1357.00**
a200-100	1346.90	1338	1344.90	1338	1344.10	1336	1343.10	1338	**1335**	1343.20	1338	**1343.10**
a300-1	2082.67	2081	2083.67	2082	2082.09	2071	2080.73	2068	2072	2081.70	**2068**	**2080.73**
a300-10	2120.10	2105	2120.00	2104	2114.30	2103	2112.00	2104	2100	2114.30	**2104**	**2112.00**
a300-100	2065.70	2060	2065.50	2060	2064.50	2058	2065.30	2054	2056	**2064.20**	**2054**	2065.30
a400-1	2782.67	2778	2780.67	2777	2777.91	2766	2777.36	2765	**2763**	**2776.80**	2765	2777.36
a400-10	2828.00	2822	2830.90	2822	2828.30	2817	2827.00	2822	**2820**	2829.10	2822	**2827.00**
a400-100	2795.70	2789	2792.90	2788	2793.10	2786	2791.30	2784	**2784**	2792.80	**2784**	**2791.30**
a500-1	3559.00	3556	3558.33	3554	3556.82	3544	3558.09	3546	**3541**	**3556.10**	3546	3558.09
a500-10	3569.00	3547	3557.70	3544	3557.20	3549	3569.00	3547	**3546**	**3557.30**	3547	3569.00
a500-100	3568.20	3553	3565.70	3554	3563.20	3555	3562.10	3552	**3547**	**3561.00**	3552	3562.10

Tabela 4.5: Comparação entre os momentos de chamada 95%, 90%, 85%, 80%, 75% e 70% em relação à qualidade das soluções alcançadas.

Ao analisar-se a Tabela 4.5, nota-se que em relação a configuração de 95%, o HM75 possui melhor desempenho ao utilizar a configuração de 90% e, usando a configuração de 85% é superior em relação a essa última. A partir desse resultado, percebe-se que o HM75 obtém soluções de melhor qualidade à medida que a chamada do resolvedor ocorre antecipadamente. Contudo, assim como não é interessante chamar o resolvedor nas primeiras iterações devido à má formação dos conjuntos E^0 e E^*, também não é interessante chamar o resolvedor nas iterações finais, pois dessa forma, pouco se contribuirá para a melhoria dos conjuntos elite.

Dessa forma, foram realizados experimentos antecipando a chamada do resolvedor, até se encontrar um momento de chamada que não resulte em melhoria de qualidade da solução final do HM75. À medida que se antecipava a chamada do resolvedor e, obtinha-se resultados melhores, era realizado um novo teste de *Friedman* usando as mesmas hipóteses e o p-valor da análise de significância estatística apresentada na Seção 4.2 (ver Tabela 4.6). Na Tabela 4.6, é apresentado para cada par de *momento_de_chamada*, e para cada grupo de três instâncias, o número de instâncias com melhores médias encontradas para cada *momento_de_chamada* e, entre parênteses, o número de vezes que ocorreu significância estatística para esses resultados.

Momento	Grupo de instâncias				
de chamada	a100	a200	a300	a400	a500
95%	0(0)	1(0)	1(0)	1(1)	0(0)
90%	3(2)	2(1)	2(2)	2(1)	3(1)
90%	0(0)	0(0)	0(0)	1(0)	0(0)
85%	3(2)	3(1)	3(1)	2(1)	3(2)
85%	2(1)	0(0)	0(0)	0(0)	2(1)
80%	1(1)	3(2)	3(2)	3(1)	1(1)
80%	1(1)	3(0)	2(0)	2(0)	0(0)
75%	2(2)	0(0)	1(1)	1(1)	3(2)
75%	1(0)	1(0)	0(0)	0(0)	2(1)
70%	2(0)	2(0)	3(0)	3(0)	1(0)

Tabela 4.6: Análise de significância estática para validação do *momento_de_chamada*.

Após os experimentos utilizando os momentos de chamada 95%, 90%, 85%, 80%, 75% e 70%, percebeu-se que a configuração de 75% foi superior a todas que a antecedem, até o momento em que é superada pela configuração de 70%. Contudo, o momento de chamada 70% despende um tempo de processamento bem maior que o momento de chamada 75% para quase todas as instâncias utilizadas, às vezes atingindo o dobro do tempo utilizado pela configuração de 75% (ver Tabela 4.7).

Na Tabela 4.7 os valores destacados em negrito representam os melhores tempos

médios de processamento e, devido aos resultados de qualidade de solução apresentados na Tabela 4.5 e de significância estatística apresentados na Tabela 4.6, os tempos de processamento que utilizaram os *momento_de_chamada* 95%, 90%, 85% e 80% não foram considerados.

Instância	75%	70%
	Tempo Médio	Tempo Médio
a100-1	**9.48**	12.71
a100-10	**9.07**	10.21
a100-100	**9.84**	10.22
a200-1	**58.30**	68.82
a200-10	**49.8**	92.58
a200-100	114.91	**55.85**
a300-1	**130.21**	210.71
a300-10	**124.82**	201.64
a300-100	**125.82**	140.6
a400-1	**228.11**	399.82
a400-10	**237.35**	415.71
a400-100	**237.24**	248.52
a500-1	**415.44**	868.68
a500-10	**401.17**	857.38
a500-100	**427.18**	433.71

Tabela 4.7: Comparação entre os momentos de chamada 75% e 70% em relação ao tempo médio de execução.

Além do tempo despendido ser bem maior ao utilizar o *momento_de_chamada* igual a 70%, os resultados apresentados na Tabela 4.6 mostram que o ganho em média de custo de solução obtido por essa configuração em relação à 75%, não possui significância estatística para nenhuma instância. Portanto, definiu-se a configuração de 75% como sendo o momento de chamada padrão para o resolvedor de PLI no método HM75.

Uma vez definido qual seria o melhor momento de invocar o resolvedor na estratégia HM75, os experimentos computacionais foram realizados. Na Tabela 4.8, são apresentados os resultados desses experimentos relacionados à qualidade de solução e, comparados com os resultados da nova proposta de otimização. Para essa comparação, cada estratégia foi executada dez vezes para cada instância, com sementes distintas. A Tabela 4.8 apresenta o custo da melhor solução obtida, o custo médio referente às dez execuções, os desvios percentuais de ambas as estratégias em relação aos critérios melhor custo de solução e média de custo, e a média desses desvios. Os melhores resultados foram destacados em negrito e os tempos médios são apresentados apenas na Tabela 4.9.

Esses resultados indicam que, ao invocar o resolvedor de PLI embutido no código do MDM-GRASP-RC, é possível melhorar a qualidade de soluções dos conjuntos elite desta heurística. Esta abordagem obteve resultados promissores em relação à proposta de pós-otimização, em termos de qualidade de solução.

Instâncias	Melhor Solução	Melhor Média	HM75 Melhor Solução	HM75 Média Solução	HM75 DP Melhor	HM75 DP Média	MPO Melhor Solução	MPO Média Solução	MPO DP Melhor	MPO DP Média
a100-1	669	677.10	**669**	**677.10**	0.000	0.000	675	679.20	0.009	0.003
a100-10	655	663.10	**655**	**663.10**	0.000	0.000	655	665.10	0.000	0.003
a100-100	659	666.20	**659**	**666.20**	0.000	0.000	661	667.20	0.003	0.002
a100-1000	640	645.30	**640**	**645.30**	0.000	0.000	641	646.30	0.002	0.002
a100-10000	658	661.10	**658**	**661.10**	0.000	0.000	658	663.50	0.000	0.004
a200-1	1365	1370.00	**1365**	**1370.00**	0.000	0.000	1367	1373.50	0.001	0.003
a200-10	1348	1359.70	**1348**	**1359.70**	0.000	0.000	1362	1363.00	0.010	0.002
a200-100	1335	1343.20	**1335**	**1343.20**	0.000	0.000	1339	1348.80	0.003	0.004
a200-1000	1345	1351.90	**1345**	**1351.90**	0.000	0.000	1349	1356.20	0.003	0.003
a200-10000	1352	1362.70	**1352**	**1362.70**	0.000	0.000	1357	1365.60	0.004	0.002
a300-1	2068	2081.70	2072	**2081.70**	0.002	0.000	**2068**	2084.80	0.000	0.001
a300-10	2100	2114.30	**2100**	**2114.30**	0.000	0.000	2113	2119.90	0.006	0.003
a300-100	2056	2064.20	**2056**	**2064.20**	0.000	0.000	2062	2067.00	0.003	0.001
a300-1000	2064	2076.00	**2064**	**2076.00**	0.000	0.000	2073	2082.90	0.004	0.003
a300-10000	2050	2061.30	**2050**	**2061.30**	0.000	0.000	2053	2064.90	0.001	0.002
a400-1	2763	2776.80	**2763**	**2776.80**	0.000	0.000	2774	2781.80	0.004	0.002
a400-10	2820	2829.10	**2820**	**2829.10**	0.000	0.000	2826	2933.80	0.002	0.037
a400-100	2784	2792.80	**2784**	**2792.80**	0.000	0.000	2791	2797.10	0.003	0.002
a400-1000	2785	2795.20	**2785**	**2795.20**	0.000	0.000	2789	2801.30	0.001	0.002
a400-10000	2812	2834.60	**2812**	**2834.60**	0.000	0.000	2821	2839.30	0.003	0.002
a500-1	3541	3556.10	**3541**	**3556.10**	0.000	0.000	3548	3561.60	0.002	0.002
a500-10	3546	3557.30	**3546**	**3557.30**	0.000	0.000	3553	3562.70	0.002	0.002
a500-100	3547	3561.00	**3547**	**3561.00**	0.000	0.000	3556	3569.40	0.003	0.002
a500-1000	3522	3541.90	**3522**	**3541.90**	0.000	0.000	3527	3546.80	0.001	0.001
a500-10000	3542	3574.80	**3542**	**3574.80**	0.000	0.000	3550	3580.20	0.002	0.002
Média					0.000	0.000			0.003	0.004

Tabela 4.8: Resultado da qualidade de solução entre as estratégias HM75 e MPO.

Na Tabela 4.9, são apresentados os resultados relacionados ao tempo de execução de ambas as estratégias. Nessa tabela, a primeira coluna representa o identificador da instância e a segunda coluna representa o melhor tempo médio obtido entre os dois métodos. A terceira e a quarta coluna mostram respectivamente o tempo médio e o seu desvio percentual em relação ao melhor tempo médio.

Na Tabela 4.9, da instância a100-1 até a instância a300-1000, o HM75 obteve tempo de execução menor que o MPO para a maioria das instâncias (perdendo somente em quatro delas), porém, para o restante das instâncias, o MPO foi superior. Em relação à

Instâncias	Melhor Tempo Médio	HM75		MPO	
		Tempo Médio	DP Tempo Médio	Tempo Médio	DP Tempo Médio
a100-1	9.48	**9.48**	0.000	9.87	0.041
a100-10	9.07	**9.07**	0.000	9.37	0.033
a100-100	9.84	**9.84**	0.000	10.27	0.044
a100-1000	8.95	**8.95**	0.000	9.54	0.066
a100-10000	9.19	**9.19**	0.000	9.71	0.057
a200-1	55.12	58.30	0.058	**55.12**	0.000
a200-10	49.80	**49.80**	0.000	55.41	0.113
a200-100	59.99	114.91	0.915	**59.99**	0.000
a200-1000	46.94	**46.94**	0.000	67.81	0.445
a200-10000	50.89	56.91	0.118	**50.89**	0.000
a300-1	130.21	**130.21**	0.000	164.93	0.267
a300-10	124.82	**124.82**	0.000	134.77	0.080
a300-100	124.37	125.82	0.012	**124.37**	0.000
a300-1000	152.47	**152.47**	0.000	276.23	0.812
a300-10000	133.02	194.90	0.465	**133.02**	0.000
a400-1	223.80	228.11	0.019	**223.80**	0.000
a400-10	234.13	237.35	0.014	**234.13**	0.000
a400-100	232.24	237.24	0.022	**232.24**	0.000
a400-1000	241.43	241.70	0.001	**241.43**	0.000
a400-10000	243.51	243.98	0.002	**243.51**	0.000
a500-1	403.76	415.44	0.029	**403.76**	0.000
a500-10	395.94	401.17	0.013	**395.94**	0.000
a500-100	424.58	427.18	0.006	**424.58**	0.000
a500-1000	420.90	422.79	0.004	**420.90**	0.000
a500-10000	414.83	423.28	0.020	**414.83**	0.000
Média			0.068		0.078

Tabela 4.9: Resultado do tempo de execução das estratégias HM75 e MPO.

média dos desvios-padrão, o MPO se afasta um pouco mais dos melhores tempos médios quando comparado ao HM75. Para uma comparação mais detalhada, nas próximas seções são apresentados testes de significância estatística, análise de comportamentos dessas estratégias e testes probabilísticos a partir de um método numérico desenvolvido por Ribeiro e Rosseti [21].

4.3.1 Significância estatística

Para validar os resultados apresentados na Tabela 4.8, assim como na Seção 4.2, foram realizados testes de *Friedman* [24] utilizando p-valor igual a 0.05 no intuito de verificar a significância estatística desses resultados. Para os resultados da Tabela 4.8, procura-se descobrir se os ganhos em qualidade de solução do HM75 foram devido à aleatoriedade

das estratégias ou não.

Em termos de qualidade de solução, assim como na Tabela 4.8, na Tabela 4.10 percebe-se que o MPO não obtém média de custo melhor que o HM75 para nenhuma instância. Para todas as instâncias em que o HM75 possui resultado melhor, 16 são com significância estatística. Nesse teste destacam-se o grupo de instâncias com $n = 500$ vértices, no qual foi possível obter significância estatística para todas as instâncias, e o grupo de instâncias com $n = 200$ vértices, que obteve significância estatística para quatro das cinco instâncias.

Estratégias	Grupo de instâncias				
	a100	a200	a300	a400	a500
HM75	5(2)	5(4)	5(2)	5(3)	5(5)
MPO	0(0)	0(0)	0(0)	0(0)	0(0)

Tabela 4.10: Análise de significância estatística para qualidade de solução.

4.3.2 Análise do comportamento das estratégias

No intuito de realizar uma investigação sobre o comportamento dessas três estratégias, utilizou-se a ferramenta *Time-to-target plots* (`TTTplots`) [3]. Nessa análise, utilizou-se essa ferramenta para o segundo método proposto neste trabalho HM75, com o mesmo alvo e instância da análise realizada anteriormente para a comparação entre MPO e MDM-GRASP-RC na Seção 4.2. Uma comparação entre essas três estratégias é ilustrada na Figura 4.2, a fim de mostrar visualmente o comportamento desses métodos quando se tem o intuito de atingir um alvo.

Na Figura 4.2, para a instância a100-1 e o alvo de valor igual a 675, pode-se perceber um deslocamento muito maior da curva que representa o HM75 em relação às curvas que representam as outras estratégias. Em 9.5 segundos, o HM75 possui probabilidade em torno de 99% de atingir o alvo, enquanto o MPO e o MDM-GRASP-RC possuem respectivamente probabilidade acerca de 60% e 34% de atingir o alvo. A maior diferença de desempenho encontra-se em 9 segundos de execução, no qual o HM75 possui aproximadamente 60%, MPO 3% e MDM-GRASP-RC 1% de atingir o alvo.

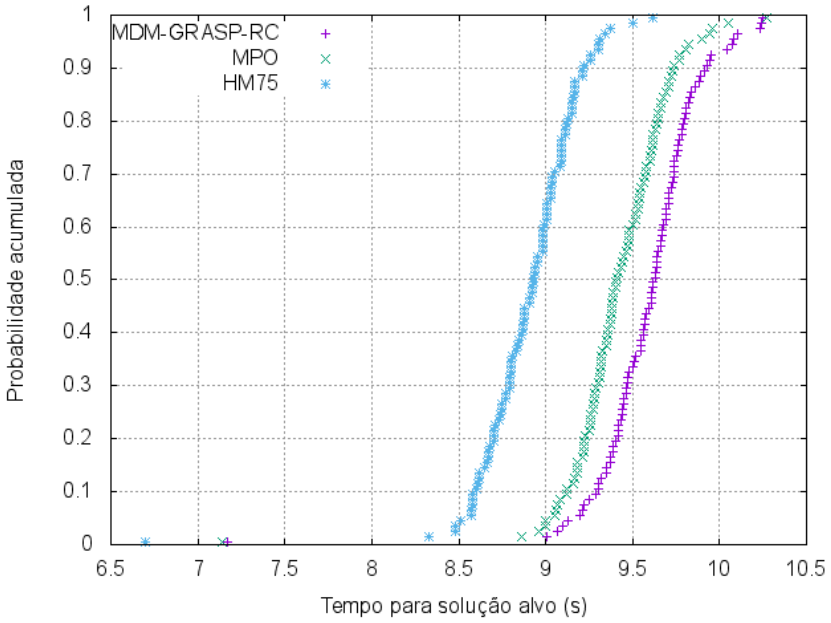


Figura 4.2: `TTTplots` para as estratégias MDM-GRASP-RC, MPO e HM75.

Para uma visão geral da diferença de custos obtidos por essas três estratégias ao longo das 25 instâncias, na Figura 4.3 é ilustrado um gráfico que apresenta o custo médio obtido por essas estratégias em cada grupo de instâncias. No eixo das abscissas, tem-se os valores que representam cada tamanho de instância e, no eixo das ordenadas (em escala logarítmica), tem-se a média de todos os custos médios das cinco instâncias de cada grupo. Na Figura 4.3, a escala logarítmica foi utilizada para ressaltar a diferença entre as médias apresentadas.

Por meio da Figura 4.3, também percebe-se que a curva que representa os custos médios do HM75, é mais próxima ao eixo das abcissas que as curvas que representam os custos médios das estratégias MDM-GRASP-RC e MPO. Para cada grupo de instâncias, as reduções em média de custo obtidas pelo HM75 em relação ao MDM-GRASP-RC, são respectivamente de 0.48%, 0.69%, 0.36%, 0.34% e 0.27%.

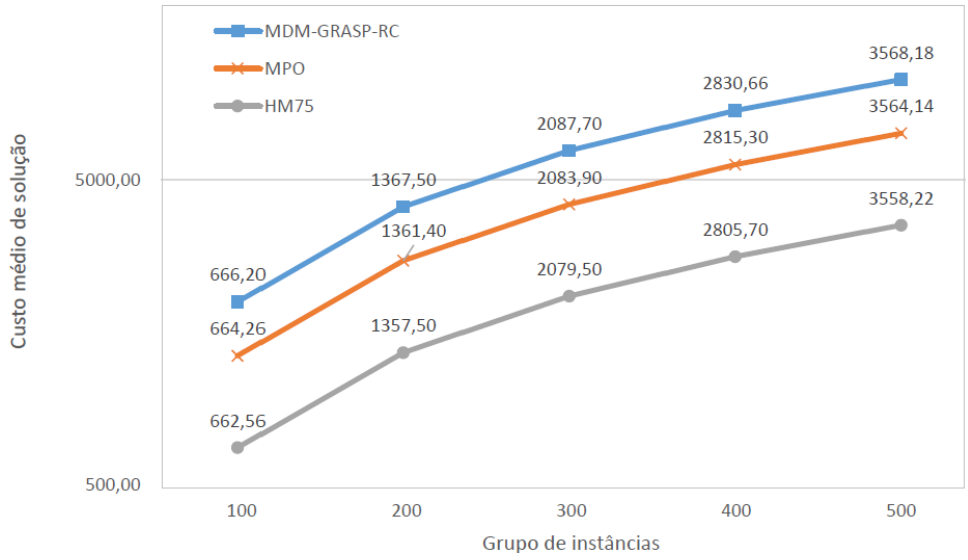


Figura 4.3: Médias de custos para as estratégias HM75, MPO e MDM-GRASP-RC.

Baseado na ferramenta `TTTplots` [3], Ribeiro e Rosseti desenvolveram em [21] um método que, dado um alvo, é possível definir a probabilidade de um algoritmo A_1 despender um tempo computacional menor que um algoritmo A_2. Trata-se de um método numérico que encontra $\Pr(A_1 < A_2)$, a probabilidade de A_1 atingir um custo de solução pelo menos tão bom quanto a um valor alvo, num tempo computacional menor que A_2. Além de um método numérico, Ribeiro e Rosseti [21] desenvolveram um programa (`TTTplots-compare`) que implementa esse método.

Por meio do uso do `TTTplots-compare` e a partir do alvo, instância e tempo de execução utilizados anteriormente nesta seção para gerar o `TTTplots` da Figura 4.2, foram realizadas comparações entre as três estratégias. Os resultados dessas comparações são apresentados na Tabela 4.11.

Na Tabela 4.11, as duas primeiras colunas identificam o algoritmos A_1 e A_2 a serem comparados, e a terceira coluna mostra a probabilidade de A_1 atingir o alvo difícil 675 antes de A_2. Para esse alvo, pode-se notar que a estratégia HM75 possui melhor desempenho em relação às outras estratégias comparadas. As probabilidades da estratégia HM75 atingir esse alvo em um tempo computacional menor que as estratégias MDM-GRASP-RC e MPO, são respectivamente 96.68% e 92.80%. Como segundo melhor método, tem-se o MPO, que possui $\Pr(A_1 < A_2)$ igual a 69.45% em relação à estratégia de Barbalho *et al.* [4].

Estratégias		$\Pr(A_1 < A_2)$
A_1	A_2	
HM75	MDM-GRASP-RC	96.68%
HM75	MPO	92.80%
MPO	MDM-GRASP-RC	69.45%

Tabela 4.11: Probabilidade de alcançar um alvo difícil em menor tempo.

Na Tabela 4.12 são apresentados os resultados desse mesmo teste, porém, utilizando um alvo intermediário de valor 679. Por meio dessa tabela, nota-se uma queda de $\Pr(A_1 < A_2)$ do método HM75 em relação aos outros dois comparados e, uma queda de $\Pr(A_1 < A_2)$ do método MPO em relação ao MDM-GRASP-RC, ambas devido a redução de dificuldade do alvo. O valor de $\Pr(A_1 < A_2)$ em relação ao MDM-GRASP-RC é de 79.42% e 78.52% em relação ao MPO. No caso do MPO, o valor de $\Pr(A_1 < A_2)$ foi reduzido para 57.76% em relação ao MDM-GRASP-RC.

Estratégias		$\Pr(A_1 < A_2)$
A_1	A_2	
HM75	MDM-GRASP-RC	79.42%
HM75	MPO	78.52%
MPO	MDM-GRASP-RC	57.76%

Tabela 4.12: Probabilidade de alcançar um alvo intermediário em menor tempo.

Este capítulo abordou os resultados experimentais obtidos pelas duas propostas apresentadas no capítulo anterior, MPO e HM75, no intuito de avaliar a contribuição da utilização de PLI aplicada à heurística MDM-GRASP-RC, em relação à qualidade de soluções obtidas e em tempo computacional despendido. Nesse sentido, as conclusões finais das análises descritas neste capítulo são apresentadas no capítulo seguinte, assim como as sugestões e as indicações de trabalhos futuros.

Capítulo 5

Conclusões

O objetivo deste trabalho foi combinar heurísticas com métodos exatos para estabelecer uma relação de cooperação e integração entre esses dois tipos de abordagem, por meio da troca de informações obtidas ao longo de suas execuções. Dessa forma, foi possível aproveitar o que cada tipo de abordagem pode oferecer de melhor para resolver o problema de síntese de redes a 2-caminhos (2-PNDP).

Dada a dificuldade de se resolver o 2-PNDP por meio de PLI pura com os modelos caminho e 2-estrela sem modificações [6], foram propostas duas combinações de métodos de PLI com a heurística estado-da-arte para esse problema (MDM-GRASP-RC) [4]. Estas combinações buscaram analisar informações obtidas durante a execução do MDM-GRASP-RC [4], e utilizá-las no intuito de acelerar o processamento do modelo de PLI. A primeira combinação, trata-se de um método de pós-otimização (MPO), no qual tenta melhorar a solução final do MDM-GRASP-RC [4] por meio de informações obtidas durante o processamento dessa heurística. A segunda combinação trata-se de um método híbrido (HM75), o qual o resolvedor de PLI busca melhorar a qualidade das soluções dos conjuntos elite do MDM-GRASP-RC [4].

Para constatar a contribuição destas novas propostas, foram realizados experimentos computacionais em 25 instâncias da literatura e os resultados indicaram o benefício dessa combinação, atingindo soluções de melhor qualidade. Atribuindo o mesmo tempo computacional para ambas as estratégias MDM-GRASP-RC [4] e MPO, essa última obteve resultados melhores para todas as 25 instâncias. Para os experimentos computacionais com o HM75, foi verificado que esse método foi capaz de aprimorar as melhores soluções e as médias obtidas pelo MPO. Portanto, esses resultados mostraram o bom desempenho da combinação de métodos heurísticos com métodos exatos para resolução do 2-PNDP.

Em seguida, análises complementares foram realizadas a fim de investigar o comportamento dessas três estratégias e confirmar a contribuição dos métodos propostos. Tais análises comprovaram que, para quase todos os casos em que as propostas deste trabalho foram superiores ao MDM-GRASP-RC [4], houve significância estatística. Essa superioridade ficou mais clara por meio das ilustrações geradas a partir da ferramenta `TTT-plots` [3], que apresentaram o desempenho desses três algoritmos quando se definiu um valor alvo de custo de solução.

Além dessas ilustrações, o método numérico `TTT-plots-compare` [21] também comprovou o melhor desempenho das estratégias HM75 e MPO, apresentando as respectivas probabilidades de 96.68% e 69.45% desses algoritmos atingirem um alvo em menor tempo computacional que o método MDM-GRASP-RC [4] para um alvo difícil. Para um alvo intermediário, as respectivas probabilidade foram de 79.42% e 57.76%.

A partir da análise das execuções da heurística estado-da-arte para o 2-PNDP, percebeu-se que essa análise pode ser aplicada a outras heurísticas que resolvem problemas de otimização combinatória. Por meio de buscas locais realizadas e soluções obtidas por essas heurísticas, os espaços de buscas dos seus respectivos problemas podem ser reduzidos.

Os seguintes tópicos podem ser considerados investigações futuras deste trabalho: (a) aplicar a técnica de *local branching* a partir do conjunto E^*, tornando o modelo caminho modificado mais flexível; (b) substituir o conjunto E^* pelo conjunto M do MDM-GRASP-RC, composto por padrões que possuem arestas com maior probabilidade de pertencerem à solução final; (c) estudar o politopo do 2-PNDP para o uso de *branch-and-cut*, buscando novas inequações válidas e, se possível, facetas; (d) aplicar a técnica de geração de colunas no 2-PNDP, considerando as soluções obtidas pelo MDM-GRASP-RC como colunas iniciais; (e) executar primeiro o resolvedor de PLI, e a partir da sua primeira solução inteira, chamar o MDM-GRASP-RC inserindo essa solução nos conjuntos M e L; (f) utilizar informações de PLI para auxiliar na construção dos conjuntos E^0 e E^*; e, por fim, (g) realizar testes invocando o resolvedor de PLI mais de uma vez no HM75.

Referências

[1] Agrawal, R., Imieliński, T., Swami, A. Mining association rules between sets of items in large databases. *ACM SIGMOD Record 22* (1993), 207–216.

[2] Agrawal, R., Srikant, R. Fast algorithms for mining association rules. Em *Proceedings. 20th International Conference on very large data bases, VLDB* (1994), vol. 1215, p. 487–499.

[3] Aiex, R. M., Resende, M. G. C., Ribeiro, C. C. TTTplots: a perl program to create time-to-target plots. *Optimization Letters 1* (2006), 355–366.

[4] Barbalho, H., Rosseti, I., Martins, S. L., Plastino, A. A hybrid data mining GRASP with path-relinking. *Computers & Operations Research 40* (2013), 3159–3173.

[5] Chaovalitwongse, W. A., Oliveira, C. A., Chiarini, B., Pardalos, P. M., Resende, M. G. C. Revised GRASP with path-relinking for the linear ordering problem. *Journal of Combinatorial Optimization 22* (2011), 572–593.

[6] Dahl, G., Johannessen, B. The 2-path network problem. *Networks 43* (2004), 190–199.

[7] Fleurent, C., Glover, F. Improved constructive multistart strategies for the quadratic assignment problem using adaptive memory. *INFORMS Journal on Computing 11* (1999), 198–204.

[8] Fürnkranz, J. Round robin rule learning. Em *Proceedings of the 18th International Conference on Machine Learning (ICML-01): 146–153* (2001), Citeseer.

[9] Glover, F. Multi-start and strategic oscillation methods and principles to exploit adaptive memory. Em *Computing Tools for Modeling, Optimization and Simulation*. Springer, 2000, p. 1–23.

[10] Glover, F., Laguna, M. *Tabu Search*. Springer, 2013.

[11] Glover, F., Laguna, M., Martí, R. Fundamentals of scatter search and path relinking. *Control and Cybernetics 29* (2000), 653–684.

[12] Goethals, B., Zaki, M. J. Advances in frequent itemset mining implementations: report on FIMI'03. *ACM SIGKDD Explorations Newsletter 6* (2004), 109–117.

[13] Han, J., Kamber, M., Pei, J. *Data mining: Concepts and techniques*. Morgan kaufmann, 2006.

[14] Han, J., Pei, J., Yin, Y. Mining frequent patterns without candidate generation. Em *ACM SIGMOD Record* (2000), vol. 29, ACM, p. 1–12.

[15] IBM. ILOG CPLEX Optimizer. Último acesso em 25/05/2015, disponível em: http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer.

[16] Laguna, M., Marti, R. GRASP and path relinking for 2-layer straight line crossing minimization. *INFORMS Journal on Computing 11* (1999), 44–52.

[17] Orlando, S., Palmerini, P., Perego, R., Silvestri, F. Adaptive and resource-aware mining of frequent sets. Em *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on* (2002), IEEE, p. 338–345.

[18] Resende, M. G. C., Ribeiro, C. C. GRASP with path-relinking: Recent advances and applications. Em *Metaheuristics: progress as real problem solvers*. Springer, 2005, p. 29–63.

[19] Ribeiro, C. C., Martins, S. L., Rosseti, I. Metaheuristics for optimization problems in computer communications. *Computer Communications 30* (2007), 656–669.

[20] Ribeiro, C. C., Rosseti, I. Efficient parallel cooperative implementations of GRASP heuristics. *Parallel Computing 33* (2007), 21–35.

[21] Ribeiro, C. C., Rosseti, I. TTTplots-compare: A perl program to compare time-to-target plots or general runtime distributions of randomized algorithms. *Optimization Letters* (2013), 1–14.

[22] Rosseti, I. *Estratégias sequenciais e paralelas de GRASP com reconexão por caminhos para o problema de síntese de redes a 2-caminhos*. Tese de Doutorado, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2003.

[23] Shapiro, S. S., Wilk, M. B. An analysis of variance test for normality (complete samples). *Biometrika* (1965), 591–611.

[24] Siegel, S., Castellan, N. J. J.(1988). Nonparametric Statistics for the Behavioral Sciences. *McGraw-HiU Book Company, New York*.

[25] Team, D. Development Core Team. R: A language and environment for statistical computing. R Foundation for Statistical Computing, 2008. Último acesso em 06/08/2015, disponível em: http://www.r-project.org.