UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE CIÊNCIAS
DEPARTAMENTO DE COMPUTAÇÃO
MESTRADO E DOUTORADO EM CIÊNCIA DA COMPUTAÇÃO

ARTHUR RODRIGUES ARARUNA

CAMINHO MÍNIMO COM RESTRIÇÃO PROBABILÍSTICA DE ATRASO MÁXIMO

FORTALEZA
2013

ARTHUR RODRIGUES ARARUNA

CAMINHO MÍNIMO COM RESTRIÇÃO PROBABILÍSTICA DE ATRASO MÁXIMO

Dissertação de Mestrado apresentada ao Programa de Mestrado e Doutorado em Ciência da Computação, do Departamento de Computação da Universidade Federal do Ceará, como requisito parcial para obtenção do Título de Mestre em Ciência da Computação. Área de concentração: Otimização (Teoria da Computação).

Orientador: Prof. Dr. Rafael Castro de Andrade

FORTALEZA
2013

Arthur Rodrigues Araruna

Caminho mínimo com restrição probabilística de atraso máximo.

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação, da Universidade Federal do Ceará, como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação na Área de Concentração Ciência da Computação

BANCA EXAMINADORA

Prof. Dr. Rafael Castro de Andrade
(Orientador)
Universidade Federal do Ceará – UFC

Prof. Dr. Manoel Bezerra Campêlo Neto
Universidade Federal do Ceará – UFC

Prof. Dr. Carlos Diego Rodrigues
Universidade Federal do Ceará – UFC

Prof. Dr. Plácido Rogério Pinheiro
Universidade de Fortaleza - UNIFOR

Fortaleza, 29 de agosto de 2013

Dedico este trabalho aos meus avós, que sempre deixaram claro aos meus pais que a maior herança que se pode deixar a um filho é a educação; à minha mãe, que se empenhou durante toda minha vida em repassar-me esse ensinamento; e ao meu pai, que mesmo não estando mais conosco, certamente se orgulharia do quanto conquistei.

AGRADECIMENTOS

Agradeço primeiramente a Deus, por tudo.

Ao meu orientador, Prof. Dr. Rafael Castro de Andrade, por auxiliar-me e amparar-me no meu desenvolvimento acadêmico, desde a minha graduação, e por toda a paciência e dedicação.

Aos membros da banca de avaliação, pelo tempo desprendido na apreciação e julgamento do trabalho.

Aos meus amigos e colegas do ParGO, na parte profissional, por terem auxiliado no desenvolvimento deste trabalho contribuindo com sugestões e questionamentos, ou mesmo apenas ouvindo tantas e tantas vezes os problemas com que me deparei; e na parte pessoal, por terem torcido por mim e promovido tantos momentos de descontração, dos quais várias vezes precisei.

À minha noiva e melhor amiga, Juliana, pela confiança, pelo apoio, pela preocupação, pela motivação, pela paciência em tanto ouvir sobre os "Caminhos Mínimos" e sobre os "$\mathcal{NP}$-completos", e por todo o amor e carinho.

Aos meus demais amigos, que próximos ou não, presentes ou não, se fizeram importantes.

À minha família, em especial à minha mãe, Vera, e minha tia, Alacoque, tendo sido tão fundamentais na minha vida, e a quem irei honrar sempre.

Finalmente à CAPES, por tornar viável a realização deste projeto através do fomento concedido.

"Ciência da Computação está tão relacionada aos computadores quanto a Astronomia aos telescópios."
(Edger Wybe Dijkstra)

RESUMO

No problema do *Caminho Mínimo com Restrição Probabilística de Atraso Máximo* visamos considerar o fator tempo no projeto de rotas de transporte de cargas em malhas viárias a custo mínimo, atentando à crescente incerteza nos tempos de percurso dessas rotas em malhas reais, e observá-lo tendo em mente estratégias de qualidade de serviço, de forma a obtermos um compromisso entre o custo de percurso e a conformidade ao prazo de chegada ao destino. Realizamos um estudo de problemas relacionados na literatura da área de otimização em redes de transporte, de forma a tentarmos conhecer melhor o problema a ser estudado, sobre o qual não tomamos conhecimento de trabalhos existentes. Desenvolvemos um esquema para enumeração de partições do espaço de soluções do problema, que utiliza uma decomposição em L para selecionar partições de forma inteligente, e que é auxiliado por soluções de relaxações do problema de forma a obter cotas para o custo ótimo. Além disso, desenvolvemos algumas estratégias de ramificação e de poda para um esquema de *Branch-and-Bound*, com uma fase de pré-processamento, de forma a tentar resolver o problema diretamente. Os resultados computacionais obtidos demonstram que somos competitivos com a ferramenta comercial utilizada para comparação em instâncias de menor porte para o problema. Para as demais instâncias, essa ferramenta se mostrou mais eficiente quanto ao tempo necessário para a resolução.

Palavras-chave: Caminho mínimo, Restrição probabilística, Otimização inteira.

ABSTRACT

In the *Probabilistic Delay Constrained Shortest Path* problem we aim to consider the time factor in the design of cargo routing paths in road networks at minimum cost, considering the increasing uncertainty in travel times of these routes in real networks, and keeping in mind strategies of quality of service, in order to obtain a compromise between the travel costs and the compliance of the arrival time at the destination. We conducted a study of related problems in the literature of transport networks optimization, in order to better understand the problem to be addressed, about which we are not aware of existing works. We developed a scheme for enumerating partitions of the solution space of this problem, which uses an L decomposition to select these partitions wisely, and is aided by solutions to relaxations of the problem to obtain bounds for the optimal cost. In addition, we developed some branching and pruning strategies for a *Branch-and-Bound* scheme, with a pre-processing phase, in order to try and solve the problem directly. The computational results show that we are competitive with the commercial tool used for comparison in the smaller instances. For the remaining instances, this tool is more efficient in the time required for solving the problem.

Keywords: Shortest path, Probabilistic constraint, Integer Optimization.

LISTA DE ABREVIATURAS E SIGLAS

B&B	*Branch and Bound*
CME-CA	Caminho Mínimo Estocástico com Custos Aleatórios *Shortest Path with Stochastic Arc-Lengths*
CME-AA	Caminho Mínimo Estocástico com Atrasos Aleatórios
CMPL	Caminho Mínimo com Peso Limitado *Weight Constrained Shortest Path* *Restricted Shortest Path*
CMRL	Caminho Mínimo com Recursos Limitados *Resource Constrained Shortest Path* *Multi-constrained Optimal Path*
CML	Caminho Multi-limitado *Multi-constrained Path*
CMR	Caminho Mínimo Robusto *Robust Shortest Path*
CMRPAM	Caminho Mínimo com Restrição Probabilística de Atraso Máximo *Probabilistic Delay Constrained Shortest Path*
LSG	Limite Superior Generalizado *Generalized Upper Bound*
QoS	Qualidade de Serviço *Quality of Service*

SUMÁRIO

1 INTRODUÇÃO

O problema do *Caminho Mínimo* e suas variações são certamente de grande importância em diversos setores da indústria e de serviços, com grande aplicação, por exemplo, no setor de transportes.

Em sua versão clássica (FORD JR., 1956), estamos interessados em encontrar, dentre um conjunto de rotas a partir de uma origem s até um destino t, aquela que possui o menor custo de percurso. Trata-se de uma versão de fácil resolução, com vários algoritmos de tempo de execução polinomial conhecidos na literatura (FORD JR., 1956; BELLMAN, 1958; DIJKSTRA, 1959).

Geralmente, na busca pelas rotas mais curtas entre a origem e o destino, consideramos como custo de percurso a distância a percorrer. Porém, deparamo-nos com certas dificuldades quando transportamos a solução deste problema para algumas aplicações da realidade, onde nem sempre uma única métrica de determinação de boas rotas é suficiente, de forma que o caminho mais curto nem sempre é a melhor escolha a ser feita. Um exemplo seria o transporte de cargas perecíveis, onde o tempo de entrega é de suma importância, e onde diversos fatores podem levar a um grande atraso na entrega e à consequente perda da carga, tais como condições das vias, atrasos burocráticos, condições sazonais, etc. Isso evidencia a necessidade de se considerar o fator tempo na escolha da melhor rota.

Há, também, outro fator de grande influência sobre rotas em malhas viárias reais: a crescente incerteza sobre os atrasos que uma determinada rota pode demandar. Quando tratamos das vias públicas de uma grande cidade, por exemplo, podemos perceber que, ao longo do dia, o tempo de deslocamento de um ponto a outro da cidade varia bastante, sendo cada vez mais difícil precisar o tempo de viagem. É necessário então tratar um certo grau de não-determinismo inerente ao problema, de forma a nos aproximarmos mais da realidade nesse tipo de aplicação.

Assim, neste trabalho, apresentamos o problema do *Caminho Mínimo com Restrição Probabilística de Atraso Máximo (CMRPAM)*, onde propomos levar em consideração simultaneamente o custo e o tempo incerto de percurso na obtenção de rotas de transporte. Além disso, aceitamos a possibilidade de uma solução nem sempre respeitar o limite de tempo preestabelecido, porém limitada por um fator de risco α, considerado em estratégias de *Qualidade de Serviço (QoS)*, de forma a obtermos um compromisso entre custo e confiabilidade nas soluções. Ao que sabemos, a ideia de considerar o não-determinismo do tempo de percurso das rotas de forma a avaliá-lo tendo em vista o provimento de *QoS* é inédita em textos da área.

O restante do texto está dividido como segue. No Capítulo 2 apresentamos uma revisão sobre textos da literatura da área que tratam de variações do problema de caminho mínimo onde podemos considerar a influência do fator tempo em suas soluções ou que têm alguma proximidade com o problema que estudamos. No Capítulo 3 introduzimos o problema CMRPAM juntamente à sua formulação matemática. No Capítulo 4 mostramos a decomposição

desenvolvida de forma a auxiliar o esquema de enumeração de partições do problema, descrito no Capítulo 5, como um seletor inteligente. O Capítulo 6 é dedicado à descrição da resolução de uma relaxação do problema através do método do subgradiente de forma a obtermos cotas inferiores melhores para os nós do esquema de enumeração. No Capítulo 7 descrevemos um esquema de *B&B* desenvolvido para resolver diretamente o problema, fazendo uso de uma fase de pré-processamento, além de variações de estratégias de ramificação e de poda dos nós para esse esquema. Reportamos os resultados computacionais dos testes realizados de forma a pôr em prática as técnicas desenvolvidas no Capítulo 8. Finalmente, fazemos as considerações finais e determinamos as perspectivas de trabalhos futuros no Capítulo 9.

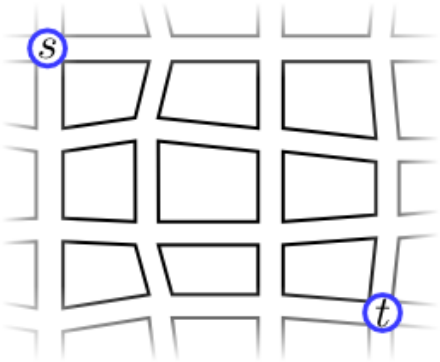
Ao longo desta dissertação, fazemos uso de conceitos e nomenclaturas das áreas de teoria de grafos e de probabilidade. Para um aprofundamento sobre esses assuntos, sugerimos a leitura de Bondy e Murty (2008) e de Ross (2009), respectivamente.

2 PROBLEMÁTICA

2.1 O Problema do Caminho Mínimo

Imagine que estamos interessados em realizar o transporte de uma dada carga a partir de um armazém até a localização definida como destino. Suponha que a Figura 1 ilustra o mapa de uma região de uma cidade onde se situam os pontos de *origem* e de *destino* da carga a ser transportada, marcados no mapa respectivamente como s e t, e considere, apenas a título de simplificação, que não desejamos trafegar por vias fora dessa região.

Figura 1 – Mapa de região de uma cidade.



Fonte: própria autoria

Seja $G = (V, E)$ o grafo que representa as vias nesse mapa, onde cada vértice corresponde a um cruzamento, e existe uma aresta entre dois vértices se, e somente se, os respectivos cruzamentos forem interligados diretamente. Defina $n := |V|$, o número de vértices, e $m := |E|$, o número de arestas desse grafo. Seja $c : E \to \mathbb{R}_+^*$ a função de custo que associa a cada aresta de G a distância entre os dois cruzamentos representados por suas extremidades. A Figura 2 exibe G, com os pontos de interesse s e t destacados, juntamente aos custos de suas arestas.

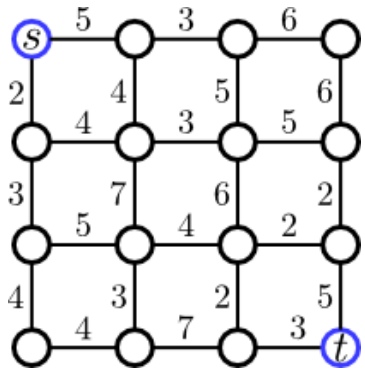
Tome um caminho qualquer entre dois vértices $u, v \in V$, que denotaremos por $u \rightsquigarrow v$. Através de um abuso de notação, expressamos o custo do caminho $u \rightsquigarrow v$ por

$$c(u \rightsquigarrow v) := \sum_{e \in u \rightsquigarrow v} c(e).$$

Representamos o conjunto de todos os caminhos entre os vértices u e v por $u \Rrightarrow v$.

O problema do *Caminho Mínimo* consiste em determinar um dentre os caminhos em $s \Rrightarrow t$ tal que seu custo seja mínimo. Em outras palavras, estamos em busca de um argumento $s \rightsquigarrow t \in s \Rrightarrow t$ que minimize a função $c(s \rightsquigarrow t)$. Ao observarmos novamente o grafo G,

Figura 2 – Grafo G e os custos de suas arestas.

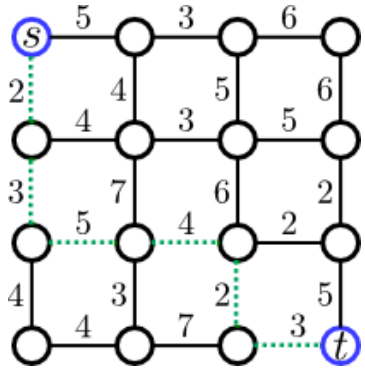


Fonte: própria autoria

Os custos das arestas verticais constam às suas esquerdas; os das horizontais, acima.

podemos perceber que o caminho $s \rightsquigarrow t$ destacado em linhas pontilhadas na Figura 3 apresenta custo mínimo, dado que nenhum outro caminho de s a t em G apresenta custo inferior a 19. $s \rightsquigarrow t$ é, portanto, uma solução para o problema.

Figura 3 – Caminho $s \rightsquigarrow t$ de custo mínimo em G.



Fonte: própria autoria

Perceba que um caminho entre quaisquer dois vértices distintos de G equivale diretamente a uma rota possível entre os respectivos cruzamentos no mapa. Dessa forma, podemos concluir que, se selecionarmos a rota no mapa que é equivalente ao caminho $s \rightsquigarrow t$ exibido, estaremos certamente transportando a carga desejada pela rota de menor distância disponível.

2.1.1 Formulação Linear

Exibimos uma formulação linear baseada em fluxos, bastante conhecida, que modela o problema do *Caminho Mínimo* e que será a base de nosso trabalho. Seja $c_{uv} := c(uv)$. Associamos duas variáveis $x_{uv}, x_{vu} \in \{0, 1\}$ a cada aresta $uv \in E$ de forma a representar se a aresta uv está sendo percorrida de u para v ou de v para u, com $u, v \in V$. Defina o conjunto $A := E \cup \{ji \mid ij \in E\}$ e considere que $c_{vu} = c_{uv} \; \forall \; uv \in E$. Representamos um dado ca-

minho $s \rightsquigarrow t$ de forma que $x_{ij} = 1$ se, e somente se, a aresta associada a x_{ij} pertence a $s \rightsquigarrow t$ e seu sentido de percurso nesse caminho é de i para j. Podemos, então, modelar o problema como segue:

$$\min \quad \sum_{uv \in A} c_{uv} x_{uv} \tag{1a}$$

$$s.a. \quad \sum_{j \mid vj \in A} x_{vj} - \sum_{i \mid iv \in A} x_{iv} = \begin{cases} 1 & \text{se } v = s \\ -1 & \text{se } v = t \\ 0 & \text{caso contrário} \end{cases} \qquad \forall\, v \in V \tag{1b}$$

$$x_{uv} \in \{0, 1\} \qquad \forall\, uv \in A \tag{1c}$$

Uma solução ótima para esse modelo linear determina uma solução para o *Caminho Mínimo*.

2.1.2 Revisão Bibliográfica

Há na literatura diversos algoritmos polinomiais que se propõem a resolver esse problema. Ford Jr. (1956) introduziu o cerne dos algoritmos para o *Caminho Mínimo* baseados em correção de rótulos[1], donde originaram-se os dois mais conhecidos, *Bellman-Ford* (BELLMAN, 1958) e *Dijkstra* (DIJKSTRA, 1959), e outro menos conhecido, *Dantzig* (DANTZIG, 1960).

Todos esses algoritmos seguem os seguintes passos. A cada vértice $v \in V$ é atribuído um rótulo $l(v)$, que indica a melhor estimativa do valor do caminho mínimo entre s e v. Inicialmente, $l(s) \leftarrow 0$ e $l(v) \leftarrow \infty\ \forall\ v \neq s$. Então, iterativamente, cada algoritmo escolhe uma aresta $uv \in E$, segundo seus critérios, tal que $l(u) + c(uv) < l(v)$. O rótulo do vértice v é então corrigido para $l(v) \leftarrow l(u) + c(uv)$. Quando não restam mais arestas a serem escolhidas, temos certeza de que $l(v)$ representa o custo de um caminho mínimo $s \rightsquigarrow v$ e, por conseguinte, $l(t)$ representa o valor da solução do problema.

O que diferencia esses algoritmos é, essencialmente, o critério de escolha do próximo rótulo a ser corrigido, o que gera impactos em suas complexidades e em seus pré-requisitos sobre o grafo dado como entrada.

Existem também outros algoritmos que se baseiam em técnicas diversas para solucionar o *Caminho Mínimo*, como os encontrados em Dantzig (1957) e Minty (1957).

2.2 Variações Determinísticas

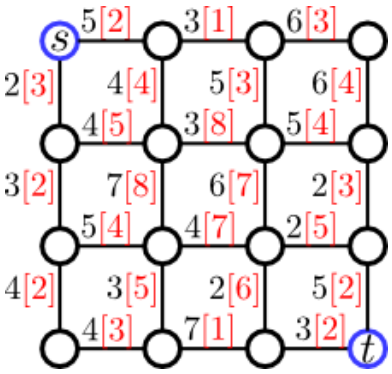
2.2.1 Caminho Mínimo com Peso Limitado

Imagine agora que firmamos o compromisso de entregar a referida carga no prazo máximo de, digamos, 20 unidades de tempo. Seja $d : E \to \mathbb{N}$ a função de atrasos que associa a

[1] N.A.: do inglês, *label correction*

cada aresta $uv \in G$ o tempo de percurso, em unidades de tempo, a que estamos sujeitos quando percorremos a quadra entre os respectivos cruzamentos no mapa. Suponha que a Figura 4 ilustre o grafo G com ambos os custos e os tempos de percurso de cada uma de suas arestas.

Figura 4 – Grafo G com ambos os custos e os atrasos de suas arestas.

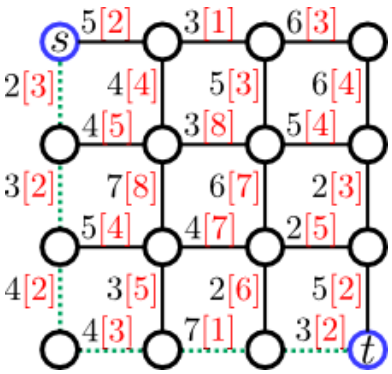


Fonte: própria autoria

Os valores das arestas verticais constam às suas esquerdas; os das horizontais, acima. Os valores entre colchetes representam o atraso de cada aresta.

Perceba que, se decidíssemos realizar o transporte da carga através da rota obtida como solução do *Caminho Mínimo*, estouraríamos o prazo de entrega, dado que o tempo de percurso dessa rota é de 24 unidades de tempo, o que poderia acarretar prejuízos caso penalidades estejam previstas ao ultrapassarmos o prazo máximo acordado. Isso evidencia que, para esse tipo de aplicação, é de extrema importância considerar também o fator tempo na busca de rotas, de forma a obter soluções satisfatórias. A Figura 5 exibe o menor caminho em G cujo tempo de percurso não excede o prazo acordado.

Figura 5 – Caminho de custo mínimo em G dentre os com tempo de percurso inferior a 20 unidades de tempo.



Fonte: própria autoria

Nesse caso, estamos interessados apenas em rotas $s \rightsquigarrow t$ tais que o tempo total de percurso é limitado. Dessas, escolheremos a de menor custo. Chamaremos esse limite

de D. Tratamos agora do problema do *Caminho Mínimo com Peso Limitado (CMPL)*, que foi introduzido pelo trabalho de Joksch (1966). Esse problema é uma variação bastante conhecida e estudada. Entretanto, ao contrário do problema original, trata-se de um problema $\mathcal{NP}$-difícil (GAREY; JOHNSON, 1979, p. 214), com aplicações, por exemplo, em gerência de linhas férreas (HALPERN; PRIESS, 1974) e em sistemas de controle aéreo militar (HIRSCH et al., 2007; ROYSET, 2009). Também aparece como subproblema no contexto do uso de geração de colunas para alguns problemas, como o escalonamento de pessoal e o roteamento de aeronaves de longa distância (BARNHART et al., 1998).

O problema de viabilidade associado ao *CMPL*, ou em outras palavras, o de determinar se existe solução para uma dada instância do problema, é *polinomial*, já que basta determinar um caminho de peso mínimo (para o caso do exemplo, tempo de percurso) para o grafo de entrada utilizando, por exemplo, uma das técnicas exibidas na seção anterior. O valor da solução desse problema será inferior ao limite D se, e somente se, a instância correspondente do *CMPL* for viável.

2.2.1.1 Formulação Linear

Podemos modificar a formulação linear apresentada para o *Caminho Mínimo* de forma a considerar apenas os caminhos com tempo de percurso limitado através da adição de uma *restrição de mochila*. Fazendo $d_{vu} = d_{uv} := d(uv)$ e sendo D o limite de tempo, o modelo a seguir representa o problema do *Caminho Mínimo com Peso Limitado*.

$$\min \quad \sum_{uv \in A} c_{uv} x_{uv} \tag{2a}$$

$$s.a. \quad \sum_{j \mid vj \in A} x_{vj} - \sum_{i \mid iv \in A} x_{iv} = \begin{cases} 1 & \text{se } v = s \\ -1 & \text{se } v = t \\ 0 & \text{caso contrário} \end{cases} \qquad \forall\, v \in V \tag{2b}$$

$$\sum_{uv \in A} d_{uv} x_{uv} \leqslant D \tag{2c}$$

$$x_{uv} \in \{0, 1\} \qquad \forall\, uv \in A \tag{2d}$$

onde a restrição (2c) é responsável por eliminar do conjunto viável todos os caminhos que tenham tempo total de percurso superior ao desejado.

2.2.1.2 Revisão Bibliográfica

Existem métodos de resolução exatos, aproximativos e heurísticos para o *CMPL* disponíveis na literatura. Seus métodos exatos são geralmente classificados entre três categorias: os que fazem uso de soluções do problema dos k *menores caminhos*, os que são baseados em programação dinâmica e rotulação de vértices e os de relaxação lagrangeana em otimização linear.

Nas soluções da primeira categoria, enumeram-se k dos menores caminhos em G com respeito a um custo definido em função dos custos das arestas c e de seus valores de atraso d, na tentativa de encontrar uma solução viável ao problema ou próxima ao ótimo. Podemos destacar os trabalhos de Skiscim e Golden (1989) e Guo e Matta (1999). Na segunda categoria, associam-se rótulos aos vértices do grafo, que são subsequentemente atualizados através de programação dinâmica. Desrosiers et al. (1995) e Jaumard, Semet e Vovor (1999) são exemplos de abordagens dessa categoria, com destaque para Desrochers e Soumis (1988). Em Widyono (1994), o autor propõe o algoritmo *Bellman-Ford Restrito*, que se baseia no algoritmo de *Bellman-Ford* (BELLMAN, 1958) para encontrar caminhos de custos monotonicamente decrescentes através da adição de novos rótulos aos vértices visitados.

Os trabalhos que se situam na última categoria são aqueles nos quais um problema dual lagrangeano é resolvido, tendo a folga (*gap*) dual eliminada utilizando-se diferentes métodos. Em Handler e Zang (1980), a folga é eliminada aplicando-se o algoritmo dos k *menores caminhos* com os custos das arestas iguais aos custos reduzidos que obtemos da solução ótima do dual lagrangeano. Mehlhorn e Ziegelmann (2000) também fazem uso dos k *menores caminhos*, apresentando também uma solução baseada em correção de rótulos. Beasley e Christofides (1989) usam os valores do dual lagrangeano como cotas inferiores para cada nó da árvore de um esquema de *B&B*. Mais recentemente, podemos encontrar o trabalho de Muhandiramge e Boland (2009) onde os autores utilizam mais de um multiplicador de Lagrange para construir o dual lagrangeano e eliminam a folga dual utilizando-se de um algoritmo de enumeração de caminhos.

Quanto aos métodos aproximativos, podemos citar Hassin (1992), que apresenta dois esquemas de aproximação completamente polinomiais distintos para a resolução do problema, aplicáveis apenas a grafos acíclicos, cujas complexidades foram ligeiramente melhoradas em Phillips (1993) e Lorenz et al. (2000). Outro trabalho nessa categoria é o de Lorenz e Raz (2001), que aprimora a complexidade do melhor dos esquemas de Hassin por um fator de n e cujo algoritmo é aplicável também para grafos que contenham ciclos. Orda (1999) apresenta um algoritmo aproximativo de melhor escalabilidade em grandes redes.

Como soluções heurísticas, encontramos os trabalhos de Sriram, Manimaran e Murthy (1998) e de Reeves e Salama (2000), que sugerem algoritmos distribuídos, e os de Guo e Matta (1999) e de Juttner et al. (2001), que fazem uso de relaxações lagrangeanas do modelo linear do *CMPL* junto a planos de corte.

2.2.2 Caminho Mínimo com Recursos Limitados

Numa generalização direta do problema do CMPL, associamos não um, mas um conjunto de k pesos (recursos) a cada aresta do grafo. Cada peso $d^i_{uv} \in \mathbb{R}$, com $uv \in E$, representa a quantidade consumida de um recurso i ao percorrermos a aresta uv em um caminho. Assumimos que dispomos de provimento limitado D_i de cada recurso i considerado, de forma que a rota pela qual desejamos transportar nossa carga não deve esgotar nenhum desses recur-

sos. Tratamos do problema do *Caminho Mínimo com Recursos Limitados (CMRL)*, que apesar de não ter relação direta com a aplicação prática que visamos com este trabalho, tem bastante aplicabilidade na resolução do CMRPAM, como veremos mais adiante. O CMRL, assim como o CMPL, foi primeiramente estudado por Joksch (1966), e também se trata de um problema $\mathcal{NP}$-difícil (JAFFE, 1984).

Ao contrário do CMPL, o problema de viabilidade do CMRL, conhecido como *Caminho Multi-limitado (CML)*, é $\mathcal{NP}$-completo (JAFFE, 1984), mas assim como sua especialização, trata-se de um problema bastante visado pela literatura.

2.2.2.1 Formulação Linear

Seja K o conjunto de recursos associados às arestas de G, $d^i_{uv} = d^i_{vu}$ a quantidade do recurso i consumida pela aresta $uv \in E$ e seja D_i a quantidade máxima do recurso i de que dispomos. Assim, podemos facilmente complementar a formulação dada para o CMPL de forma a modelar o problema CMRL, como segue:

$$\min \quad \sum_{uv \in A} c_{uv} x_{uv} \tag{3a}$$

$$s.a. \quad \sum_{j \mid vj \in A} x_{vj} - \sum_{i \mid iv \in A} x_{iv} = \begin{cases} 1 & \text{se } v = s \\ -1 & \text{se } v = t \\ 0 & \text{caso contrário} \end{cases} \qquad \forall v \in V \tag{3b}$$

$$\sum_{uv \in A} d^k_{uv} x_{uv} \leqslant D_k \qquad \forall k \in K \tag{3c}$$

$$x_{uv} \in \{0, 1\} \qquad \forall uv \in A \tag{3d}$$

2.2.2.2 Revisão Bibliográfica

CMRL

O problema do CMRL tem aplicações em redes de telecomunicações (XUE, 2000; NYGAARD; MELNIKOV; KATSAGGELOS, 2001; MIEGHEM et al., 2003), em engenharia civil (ELIMAM; KOHLER, 1997), aviação militar (ZABARANKIN; URYASEV; PARDALOS, 2002) e computação gráfica (NYGAARD, 2000). Também aparece no contexto de geração de colunas e *Branch-and-Price* na resolução de outros problemas bem conhecidos, tais como o *Roteamento de Veículos com Janelas de Tempo*[2] (DESROCHERS; DESROSIERS; SOLOMON, 1992), gerência de frotas de veículos (AVELLA; BOCCIA; SFORZA, 2004) e problemas relacionados a serviços de voos urgentes[3] (ESPINOZA et al., 2008), por exemplo.

Devido à sua intrínseca conexão com o CMPL, diversos trabalhos encontrados abordam os dois problemas em conjunto, como os já mencionados de Handler e Zang (1980), Des-

[2] N.A.: do inglês, *Vehicle Routing with Time Windows*

[3] N.A.: numa tradução livre do inglês *dial-a-flight*

rochers e Soumis (1988), Beasley e Christofides (1989), Skiscim e Golden (1989), Desrosiers et al. (1995), Jaumard, Semet e Vovor (1999) e Mehlhorn e Ziegelmann (2000). Dentre os trabalhos que focam no CMRL, destacamos os trabalhos de Boland, Dethridge e Dumitrescu (2006) e de Righini e Salani (2008), baseados na solução proposta por Kohl (1995), que é considerada a abordagem mais eficiente ao problema (PUGLIESE; GUERRIERO, 2012a).

Em Ziegelmann (2007) o autor propõe uma solução utilizando a envoltória convexa do problema, considerando três métodos distintos para eliminar a folga entre as cotas obtidas: o algoritmo aproximativo de Hassin (1992), o algoritmo dos k *menores caminhos* de Jiménez e Marzal (1999) e um algoritmo de programação dinâmica. Nos trabalhos de Avella, Boccia e Sforza (2002) e Avella, Boccia e Sforza (2004), os autores apresentam funções de penalização exponencial a serem utilizadas em relaxações lagrangeanas do problema e conduzem testes computacionais comparativos com outras soluções. Dumitrescu e Boland (2003) descrevem um algoritmo de rotulação[4] de vértices, baseado no trabalho de Desrochers e Soumis (1988), que faz uso de informações de penalizadores de Lagrange para determinar soluções viáveis e eliminar folgas entre as cotas obtidas, além de fazerem uso de uma fase de pré-processamento de forma a diminuir o número de arestas no grafo.

Em Carlyle, Royset e Wood (2008) os autores dualizam as restrições dos recursos, resolvem a relaxação obtida e eliminam folgas de otimalidade enumerando *Caminhos Mínimos Quase Ótimos*, ou ϵ-ótimos, que consistem em caminhos cujo custo é no máximo ϵ vezes o custo de um caminho mínimo. Nesse trabalho, os custos das arestas são os custos dualizados e o valor de ϵ é o valor da folga obtida até o momento.

Por fim, no trabalho de Pugliese e Guerriero (2012b), um método iterativo é apresentado usando uma metodologia diferente de busca no espaço de soluções do problema, baseada na abordagem de Wierzbicki, Makowski e Wessels (2000), chamada de *Abordagem de Ponto de Referência*[5].

CML

As soluções propostas para a resolução do CML dividem-se em quatro categorias: algoritmos aproximativos, aleatórios, heurísticos e exatos. Os algoritmos heurísticos e os aproximativos sugeridos para o CMRL são também utilizados para resolver o CML, dado que quaisquer soluções obtidas configuram certificados de viabilidade desse problema.

Além desses, podemos encontrar algoritmos aproximativos como o de Jaffe (1984), que trabalha com custos modificados em uma combinação linear entre os custos originais e os pesos das arestas, e como o de Chen e Nahrstedt (1998), que transforma a instância do CML na de um problema mais simples, cujo espaço de soluções está contido propriamente no deste.

Como exemplo de algoritmo aleatório, podemos citar o de Korkmaz e Krunz (2001b), que busca caminhos através de uma busca em largura aleatorizada, fazendo uso de informações

[4] N.A.: do inglês, *label-setting*

[5] N.A.: numa tradução livre do inglês *Reference Point Approach*

de caminhos mínimos entre os vértices do grafo previamente computadas. No campo das heurísticas, existe a solução de Neve e Mieghem (2000), conhecida como TAMCRA, onde uma métrica não-linear é aplicada aos caminhos do grafo, e estes são enumerados utilizando-se algoritmos de k *menores caminhos*. Há também o algoritmo H_MCOP (KORKMAZ; KRUNZ, 2001a), baseado no TAMCRA, onde a busca por soluções é feita partindo simultaneamente da origem e do destino no grafo através de duas variações do algoritmo de Dijkstra. Em Yuan (2002) são apresentadas duas heurísticas baseadas no algoritmo de Bellman-Ford e que aproveitam alguns conceitos do algoritmo TAMCRA.

Por último, temos a abordagem exata proposta por Mieghem, Neve e Kuipers (2001), chamada SAMCRA, bastante próxima do algoritmo TAMCRA, entretanto permitindo o aproveitamento de diferentes métricas aos caminhos e introduzindo a diferenciação por dominância, de forma a evitar o armazenamento de caminhos que certamente não levarão a soluções para o problema, permitindo a redução do espaço de busca sem comprometer a obtenção da solução.

2.3 Variações Estocásticas

2.3.1 Caminho Mínimo Estocástico com Custos Aleatórios

Os problemas de *Caminho Mínimo Estocástico com Custos Aleatórios (CME-CA)* são aplicáveis em situações onde não podemos precisar os custos de percurso das arestas, por conta de sua natureza mutável, por exemplo, mas podemos aproximá-los por variáveis aleatórias de distribuição de probabilidade conhecida. Nesse tipo de aplicação, não podemos associar apenas um valor de custo a cada aresta sob o risco de obter soluções que, devido à natureza incerta, quando levadas à pratica serão extremamente ineficientes.

A cada aresta de $uv \in G$ é associada uma variável aleatória δ_{uv} que descreve os possíveis custos da aresta e estamos em busca de um caminho $s \rightsquigarrow t$ que seja mínimo segundo um critério de otimalidade, a melhor escolha dentro de uma margem de confiança ou que minimize o valor da esperança de seu custo.

2.3.1.1 Revisão Bibliográfica

Os problemas de CME-CA são relativamente pouco estudados na literatura. Entretanto, podemos encontrar alguns trabalhos interessantes que propõem metodologias, algoritmos, modelos ou critérios para essa classe de problemas. Não conseguimos, entretanto, encontrar referências que discorressem sobre a dificuldade computacional desses problemas.

O artigo de Trevizan e Veloso (2013) apresenta uma aplicação de problemas de CME-CA na área de Inteligência Artificial, de forma a auxiliar a movimentação de um agente autônomo em um ambiente com o objetivo de buscar determinado objeto minimizando o custo de movimentação. Em Ji (2005) são estudados três critérios de otimalidade para problemas

de CME-CA, denominados *caminho mínimo esperado*[6], *caminho mínimo mais provável*[7] e α-*caminho mínimo*[8], juntamente a exemplos de situações onde cada critério poderia ser empregado.

Para um caminho $s \rightsquigarrow t$, se sua esperança de custo é mínima, dizemos que se trata de um **caminho mínimo esperado**; se a probabilidade de o custo de $s \rightsquigarrow t$ ser menor que um valor predeterminado T_0 é maior que a probabilidade dos custos dos demais caminhos em $s \Rrightarrow t$ serem menores que esse mesmo valor, dizemos que se trata de um **caminho mínimo mais provável**; e, dada uma margem probabilística predeterminada α e o menor valor $\bar{T}_{s \rightsquigarrow t}$ tal que a probabilidade de o custo de $s \rightsquigarrow t$ ser no máximo $\bar{T}_{s \rightsquigarrow t}$ é pelo menos α, se o valor de $\bar{T}_{s \rightsquigarrow t}$ for mínimo dentre todos os caminhos em $s \Rrightarrow t$, dizemos que se trata de um **α-caminho mínimo**.

Ainda em Ji (2005), também é apresentado um algoritmo que combina conceitos de algoritmos genéticos e simulação estocástica de forma a obter soluções para os problemas propostos, bem como alguns resultados de testes computacionais realizados.

Em Mirchandani (1976) é estudado o problema no contexto da busca pelo caminho que provê a menor esperança de custo, apresentando um algoritmo para resolução do problema quando as variáveis aleatórias associadas às arestas seguem distribuições discretas. No trabalho Sigal, Pritsker e Solberg (1980) é introduzido o conceito de *índice de otimalidade*, definido como a probabilidade de um caminho $s \rightsquigarrow t$ ser menor que todos os outros caminhos em $s \Rrightarrow t$, e desenvolve o conceito em grafos direcionados e acíclicos. No artigo Polychronopoulos e Tsitsiklis (1996) os autores fazem a suposição de que informações sobre o custo das arestas podem ser acumuladas durante um percurso no grafo, essencialmente significando que o custo de cada aresta é "determinado" no momento em que a aresta é percorrida, e apresentam algoritmos heurísticos de programação dinâmica para encontrar soluções de custo esperado mínimo.

Em Alexopoulos (1997), podemos encontrar métodos de resolução dos problemas que usam os critérios de caminho mínimo mais provável e α-caminho mínimo. Os métodos são baseados em partições iterativas do espaço de estados do grafo (espaço contendo todas as possíveis valorações dos custos das arestas) e geram limites que são melhorados a cada iteração. Os autores reportam resultados promissores sobre instâncias que se mostram complicadoras para outros métodos existentes à época.

2.3.2 Caminho Mínimo Estocástico com Atrasos Aleatórios

Os problemas de *Caminho Mínimo Estocástico com Atrasos Aleatórios (CME-AA)* são, dentre os problemas neste capítulo, os que mais se assemelham ao problema do CMRPAM no contexto de suas aplicações práticas. Em todos esses problemas, nos deparamos com grafos com arestas associadas a dois tipos de pesos: um determinístico, que desejamos minimizar, e

[6] N.A.: do inglês, *expected shortest path*
[7] N.A.: adaptado do inglês, *most shortest path*
[8] N.A.: do inglês, α*-shortest path*

outro estocástico, que desejamos limitar. A diferença está essencialmente na maneira como tratamos o não-determinismo da restrição de atraso máximo, como veremos no Capítulo 3.

Esses problemas parecem não ter tido muita atenção até o momento, mas podemos encontrar o trabalho de Verweij et al. (2003), que demonstra tratarem-se de problemas $\mathcal{NP}$-difíceis quando assumimos distribuições discretas dos valores de atraso.

2.3.2.1 Formulação Linear

$$\min \quad \sum_{uv \in A} c_{uv} x_{uv} + \max \left\{ \mathsf{d} \cdot \mathbb{E}(\sum_{uv \in A} \xi_{uv} x_{uv} - D);\ 0 \right\} \tag{4a}$$

$$s.a. \quad \sum_{j \mid vj \in A} x_{vj} - \sum_{i \mid iv \in A} x_{iv} = \begin{cases} 1 & \text{se } v = s \\ -1 & \text{se } v = t \\ 0 & \text{caso contrário} \end{cases} \quad \forall v \in V \tag{4b}$$

$$x_{uv} \in \{0, 1\} \qquad \forall\ uv \in A \tag{4c}$$

onde $\max \left\{ \mathsf{d} \cdot \mathbb{E}(\sum_{uv \in A} \xi_{uv} x_{uv} - D);\ 0 \right\}$ determina a penalização imposta, descrita a seguir.

2.3.2.2 Revisão Bibliográfica

Encontramos dois trabalhos (KOSUCH; LISSER, 2010; CHENG; KOSUCH; LISSER, 2012) abordando esse tipo de versão do Caminho Mínimo e que assumem que os valores dos atrasos obedecem à distribuição Normal. Em ambos, os autores consideram que a violação do limite de tempo desejado incorre diretamente em penalidades às soluções, que são consideradas como ônus. Nesses trabalhos, busca-se encontrar um caminho que minimize a soma de seu custo de percurso e o ônus esperado. As penalidades são devidas apenas em caso de excedente e por cada unidade de tempo esperada a mais que o máximo desejado, de forma que, supondo que d representa o custo de penalidade unitária, D representa o limite máximo de atraso desejado e ξ_{uv} representa a variável aleatória normal associada aos atrasos de $uv \in E$, temos que a penalidade aplicada a um caminho $s \rightsquigarrow t$ é calculada como

$$p(s \rightsquigarrow t) := \max\{\mathsf{d} \cdot \left(\mathbb{E}(\sum_{uv\ \in\ s \rightsquigarrow t} \xi_{uv}) - D \right);\ 0\},$$

sendo $\mathbb{E}(X)$ a notação que denota a esperança de uma variável aleatória X.

A ideia no primeiro artigo, Kosuch e Lisser (2010), é realizar uma busca no espaço de caminhos $s \Rrightarrow t$ através de um *Branch-and-Bound* com esquema de ramificação especializado. Porções do espaço que não contêm a solução ótima são eliminadas resolvendo-se uma

relaxação do problema pelo método do *Gradiente Estocástico Projetado*[9] e podando-se nós da árvore de busca a partir das soluções relaxadas obtidas. Nesse trabalho, os autores descrevem a estratégia de poda e o algoritmo de Gradiente Estocástico Projetado utilizado para resolver os nós da árvore de B&B.

No segundo, Cheng, Kosuch e Lisser (2012), que continua o trabalho desenvolvido no primeiro, os autores demonstram que, supondo que a média e a variância dos atrasos de cada arco sejam respectivamente iguais ao custo desse arco multiplicado por constantes positivas, o caminho mínimo clássico sob os mesmos custos é uma solução ótima para esse problema. Os autores exibem uma formulação determinística equivalente à formulação estocástica do problema, válida por conta da suposição da distribuição Normal dos atrasos das arestas, e exibem comparativos entre a resolução desse modelo determinístico e sua resolução do modelo estocástico.

A seguinte formulação matemática modela o problema tratado nos artigos descritos nesta seção.

2.3.3 Caminho Mínimo Robusto

Problemas de *Caminho Mínimo Robusto (CMR)* surgem no panorama de aplicações onde os custos de percurso das arestas são incertos, porém, ao contrário dos problemas de CMECA, não se pode descrevê-los eficientemente sob a regência de leis de probabilidade, ou mesmo quando conhecemos o conjunto de valoração dos custos mas não sabemos a probabilidade com que cada valor pode se apresentar. Sendo assim, surge a necessidade de se desenvolver *modelos de representação* das incertezas nos custos das arestas, que constituem uma maneira de descrever os possíveis valores de custo de cada aresta.

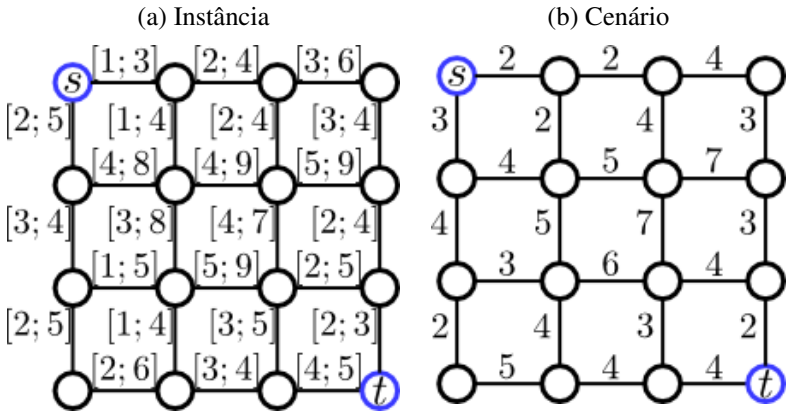
Dois modelos são mais proeminentes na literatura: o *modelo de intervalo* e o *modelo de conjunto discreto*. Em cada um desses modelos associamos conjuntos de valores a cada aresta e assumimos que o custo real da aresta $ij \in E$ reside no seu conjunto associado. No primeiro modelo, a cada aresta ij associamos intervalos da forma $[l_{ij}; u_{ij}]$. No segundo modelo, são associados conjuntos discretos finitos C_{ij}.

A partir do modelo escolhido, construímos o que chamamos de *cenário*. Um cenário nada mais é que uma configuração de custos para todas as arestas, sendo uma "foto" do grafo, onde cada aresta figura com um valor de custo dentre os seus valores disponíveis. Sendo assim, a cada cenário k, a aresta ij figura com um custo $c^k_{ij} \in [l_{ij}; u_{ij}]$, no caso do primeiro modelo, ou com $c^k_{ij} \in C_{ij}$, no caso do segundo. O conjunto de todos os cenários possíveis, ou seja, todas as combinações de valoração de arestas do grafo, é denotado por K. As Figuras 6 e 7 ilustram respectivamente instâncias do CMR sob o modelo de intervalo e de conjunto discreto juntamente a um de seus possíveis cenários.

Nos problemas de CMR, as soluções são escolhidas baseando-se em *critérios de robustez*, que constituem formas de analisar as soluções em seus cenários pessimistas de forma

[9] N.A.: do inglês: *Projected Stochastic Gradient*

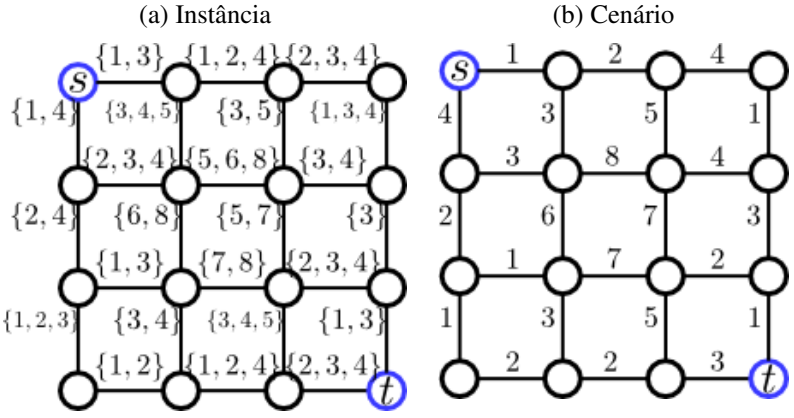
Figura 6 – Instância de CMR sob o modelo de intervalo e exemplo de um dos possíveis cenários.



Fonte: própria autoria

Os valores das arestas verticais constam às suas esquerdas; os das horizontais, acima.

Figura 7 – Instância de CMR sob o modelo de conjunto discreto e exemplo de um dos possíveis cenários.



Fonte: própria autoria

Os valores das arestas verticais constam às suas esquerdas; os das horizontais, acima.

a tentar diminuir o impacto dos seus piores casos de custo. Em outras palavras, estamos buscando soluções que são aceitáveis na grande maioria dos casos, mas que nunca serão péssimas escolhas, dentro do conhecimento que possuímos com relação aos custos das arestas provido pelo modelo de representação escolhido.

Os dois critérios de robustez mais estudados e aplicados a caminhos são os conhecidos como *robustez absoluta* (ou *critério do pior caso*) e *desvio de robustez* (ou *critério do máximo arrependimento*). No primeiro deles, buscamos o caminho de menor (dentre todos os caminhos em $s \Rrightarrow t$) máximo custo (dentre todos os cenários possíveis $k \in K$). No segundo, chamamos de *desvio de robustez* de um caminho $s \rightsquigarrow t$ a maior diferença entre seu custo em um cenário k e o de um caminho mínimo em G nesse mesmo cenário, para todos os cenários $k \in K$. Buscamos, então, um caminho que possua desvio de robustez mínimo.

A combinação de escolhas entre modelos de representação e critérios de robustez define um problema de CMR.

2.3.3.1 Formulações Lineares

A seguir, mostramos formulações lineares para problemas de CMR sob os dois critérios robustos que descrevemos.

Robustez absoluta

$$\min_{x} \max_{k \in K} \quad \sum_{uv \in A} c^k_{uv} x_{uv} \tag{5a}$$

$$s.a. \quad \sum_{j \mid vj \in A} x_{vj} - \sum_{i \mid iv \in A} x_{iv} = \begin{cases} 1 & \text{se } v = s \\ -1 & \text{se } v = t \\ 0 & \text{caso contrário} \end{cases} \qquad \forall\, v \in V \tag{5b}$$

$$x_{uv} \in \{0, 1\} \qquad \forall\, uv \in A \tag{5c}$$

Desvio de robustez

$$\min_{x} \left(\max_{k \in K} \quad \sum_{uv \in E} (c^k_{uv} x_{uv}) - \bar{c}^k \right) \tag{6a}$$

$$s.a. \quad \sum_{j \mid vj \in A} x_{vj} - \sum_{i \mid iv \in A} x_{iv} = \begin{cases} 1 & \text{se } v = s \\ -1 & \text{se } v = t \\ 0 & \text{caso contrário} \end{cases} \qquad \forall\, v \in V \tag{6b}$$

$$x_{uv} \in \{0, 1\} \qquad \forall\, uv \in A \tag{6c}$$

onde a variável $\bar{c}^k$ representa o custo de um caminho mínimo em G no cenário k.

2.3.3.2 Revisão Bibliográfica

Podemos encontrar em Yu e Yang (1998), Averbakh (2001) e Zieliński (2004) demosntrações de que os problemas de CMR sob o critério de desvio de robustez são $\mathcal{NP}$-difíceis. Em Yu e Yang (1998), o autor demonstra que mesmo para grafos bastante restritos conhecidos como *redes em camadas*[10], descritos em seu trabalho, a dificuldade computacional permanece. Para o caso de problemas sob o critério de robustez absoluta, o autor demonstra que quando o número de cenários não é limitado, esses problemas são também $\mathcal{NP}$-difíceis, entretanto, sabemos que para os modelos de intervalo e de conjunto discreto, neste último no caso em que os conjuntos possuem um elemento máximo, podemos resolver os problemas associados com algoritmos polinomiais (KARASAN; PINAR; YAMAN, 2001). Yu e Yang (1998) também apresentam algoritmos pseudo-polinomiais e heurísticos para a resolução de problemas de CMR.

Em Karasan, Pinar e Yaman (2001), Montemanni e Gambardella (2004), Montemanni, Gambardella e Donati (2004), Montemanni e Gambardella (2005) e Montemanni e Gambardella (2008) são apresentadas soluções exatas para esses problemas. Os autores de Gabrel e Murat (2007) apresentam uma boa explanação sobre problemas de CMR e seus métodos de resolução.

Por fim, mais recentemente em Gabrel, Murat e Wu (2011) os autores apresentam formulações lineares para um problema de CMR sob um novo critério de robustez introduzido por Roy (2010), conhecido como bw-*robustez*. Testes computacionais e a demonstração de sua $\mathcal{NP}$-dificuldade para o modelo de intervalo são exibidos nesse trabalho.

2.4 Outras Variações

Há ainda outras variações do problema do Caminho Mínimo pouco estudadas, mas que têm aplicações práticas em áreas bastante próximas às das aplicações que focamos no caso do CMRPAM, como por exemplo o *Caminho Mínimo com Recursos Limitados por Janelas* (IRNICH; DESAULNIERS, 2005) e o *Caminho Mínimo com Incertezas* (HERNANDES, 2007). As referências anexadas apresentam os referidos problemas junto a seus estados-da-arte.

[10] N.A.: do inglês: *layered networks*

3 CAMINHO MÍNIMO COM RESTRIÇÃO PROBABILÍSTICA DE ATRASO MÁXIMO

3.1 Preliminares

Descrevemos os problemas no Capítulo 2 baseando-nos em malhas simples quaisquer, de forma a facilitar a exibição de exemplos de instâncias. Entretanto, para o problema que tratamos neste trabalho, restringimo-nos a malhas direcionadas, para nos permitir uma representação mais acurada de malhas viárias reais, onde dois tipos de vias podem ser encontradas: as de mão-única e as de mão-dupla. Com grafos não-direcionados não conseguimos expressar os sentidos não-permitidos de percurso em uma aresta, o que compromete a representação do primeiro tipo de via. Todavia, essa mudança não prejudica a expressividade das instâncias modeladas, pois cada aresta não-direcionada uv pode ser expressa por outras duas direcionadas, uma de u para v e outra de v para u, ambas com os mesmos custos e atrasos que uv.

Dessa forma, consideramos G como sendo um grafo direcionado, com $G = (N, A)$ onde N é o conjunto de *nós* (vértices) e A é o conjunto de *arcos* (arestas direcionadas), sendo $|N| = n$ e $|A| = m$.

3.2 Motivação

Neste trabalho, estamos interessados em considerar o tempo necessário para a travessia das rotas disponíveis, que chamamos de *atraso*, como critério adicional na escolha da melhor rota a ser utilizada. Supomos que nos é imposto um prazo limite de chegada ao destino, que denotamos por D, que dita o tempo máximo que podemos desprender no percurso da rota escolhida. Essa é a situação encontrada em vários tipos de aplicações de roteamento de cargas, por exemplo. Podemos citar o caso de empresas de transporte de cargas perecíveis, ou de outros tipos de transportadoras que assumem compromissos quanto ao prazo de entrega sob pena de incorrer em multas por atraso.

A crescente incerteza sobre os valores dos tempos de percurso das rotas disponíveis, cuja causa pode ser atribuída a condições das vias, número de veículos em trânsito, condições sazonais, atrasos burocráticos ou aduaneiros, etc., impossibilita-nos de obter soluções confiáveis, quando transportadas para a prática, se nos basearmos em valores determinísticos para esse custo. Muitas vezes essas soluções não satisfazem seu propósito e levam a ônus indesejados. Para lidar com essa situação, podemos supor que, apesar de incertos, os valores de atraso obedecem uma dada distribuição de probabilidade de parâmetros conhecidos, de forma que esses valores possam ser obtidos por meio de amostras aleatórias dessa distribuição.

Consideramos, dessa forma, o problema de determinar o menor caminho de s a t tal que a probabilidade de o atraso desse caminho exceder o valor do prazo limite seja limitada por um fator de risco α, que projetistas de rotas geralmente consideram em suas estratégias de

QoS. Chamamos esse problema de *Caminho Mínimo com Restrição Probabilística de Atraso Máximo (CMRPAM)*.

Tratar com tal restrição probabilística usando ferramentas de Otimização Inteira convencionais não é trivial. Por isso, propomos a adoção de um modelo determinístico equivalente, obtido através do uso de uma técnica apresentada recentemente com o intuito de tratar de modelos estocásticos com restrições probabilísticas tais como a que lidamos. Exibimos, a seguir, o modelo não-determinístico que descreve o problema do CMRPAM e, logo após, descrevemos como podemos obter a versão determinística correspondente.

3.3 Formulação Matemática

3.3.1 Versão Não-Determinística

Seja ξ o parâmetro aleatório que representa o conjunto de atrasos de cada arco e suponha que conhecemos a sua Função de Distribuição de Probabilidade[1]. Denotaremos por d^{ξ}_{uv} o atraso independente associado ao arco uv regido por ξ. Chamamos de *cenário* uma coleção $\{d^{\xi}_{uv} \mid uv \in A\}$ de amostragens do parâmetro ξ para os arcos de G.

Seja $u \rightsquigarrow v$ um caminho em G. Dizemos que o *custo de percurso* de $u \rightsquigarrow v$, denotado por $c(u \rightsquigarrow v)$, é o somatório dos custos de cada arco pertencente ao dado caminho. Ou seja,

$$c(u \rightsquigarrow v) := \sum_{ij \,\in\, u \rightsquigarrow v} c_{ij},$$

onde c_{ij} denota o custo do arco $ij \in A$. Além disso, dizemos que o *atraso de percurso* de $u \rightsquigarrow v$, denotado por $d^{\xi}(u \rightsquigarrow v)$, é o somatório dos atrasos de cada arco pertencente ao dado caminho. Ou seja,

$$d^{\xi}(u \rightsquigarrow v) := \sum_{ij \,\in\, u \rightsquigarrow v} d^{\xi}_{ij}$$

Denote por $D \in \mathbb{N}$ o limite de atraso de percurso que desejamos impor sobre a solução do problema e seja $\alpha \in [0; 1]$ a margem de risco probabilística que estamos dispostos a correr sobre a não conformidade da solução em relação ao limite de atraso citado. Se representarmos um dado caminho $u \rightsquigarrow v$ em G através de um vetor $x \in \{0, 1\}^{|A|}$, indexado pelos arcos do grafo, tal que $x_{ij} = 1$ se, e somente se, $ij \in u \rightsquigarrow v$ (vetor esse que chamaremos de *vetor característico*), podemos escrever a seguinte formulação não-determinística para o problema CMRPAM:

[1] Para uma visualização de como são tratados os atrasos dos arcos da rede, veja no Apêndice A a descrição das nossas instâncias.

$$\min \quad \sum_{uv \in A} c_{uv} x_{uv} \tag{7a}$$

$$s.a. \quad \sum_{j \mid vj \in A} x_{vj} - \sum_{i \mid iv \in A} x_{iv} = \begin{cases} 1 & \text{se } v = s \\ -1 & \text{se } v = t \\ 0 & \text{caso contrário} \end{cases} \qquad \forall\, v \in N \tag{7b}$$

$$\mathbb{P}(\sum_{uv \in A} d^{\xi}_{uv} x_{uv} > D) \leqslant \alpha \tag{7c}$$

$$x \in \{0,1\}^{|A|} \tag{7d}$$

Na restrição probabilística (7c) representamos o limite de risco de o atraso de percurso da solução ser superior ao valor D que impomos às soluções, dado que $\mathbb{P}(\sum_{uv \in A} d^{\xi}_{uv} x_{uv} > D)$ representa a probabilidade de ocorrência dessa violação, não podendo ultrapassar $\alpha \in [0; 1]$.

A seguir, mostramos como obter uma versão linear determinística para a formulação apresentada, de forma a nos permitir aplicar técnicas de otimização linear para a resolução do problema.

3.3.2 Versão Determinística

Neste trabalho, medimos o atraso de percurso dos caminhos a considerar em unidades de tempo. Por isso, assumimos que o parâmetro aleatório que rege os atrasos dos arcos no grafo G segue uma distribuição de probabilidade de valores discretos. Essa escolha é justificada pela possibilidade de limitar o número de possíveis cenários de atraso da rede. Note que não precisamos considerar amostras de ξ onde $d^{\xi}_{uv} \geqslant D$ para algum $uv \in A \setminus \{st\}$ pertencendo a algum caminho $s \rightsquigarrow t \in s \Rrightarrow t$, dado que a partir desse valor tal caminho já não fará parte do espaço de soluções viáveis do problema, não mais influenciando a escolha da solução ótima. Os valores das amostras para arcos que não pertencem a nenhum caminho $s \rightsquigarrow t$ em G não afetam a solução do problema, o que nos permite, sem perda de generalidade, desconsiderar atrasos superiores a D também para esses arcos.

Nesse caso, fica evidente que só precisamos considerar um conjunto finito de amostras de ξ. Em outras palavras, nosso conjunto K de cenários é finito.

Note, agora, que o lado direito da inequação dentro da notação de probabilidade na restrição (7c) é constante. Portanto, para uma dada amostragem fixa do parâmetro aleatório, o conjunto definido por essa restrição é convexo. Tudo isso indica que podemos utilizar o conjunto finito K de cenários de ξ de forma a linearizar a restrição (7c) utilizando a técnica apresentada recentemente em **??**).

Tal técnica consiste em substituir a restrição probabilística por uma restrição de mochila para cada cenário, onde os custos dos arcos são dados pelos valores dos atrasos dos respectivos cenários. Além disso, de forma a modelar o limite probabilístico, introduzimos

variáveis $z \in \{0,1\}^{|K|}$, onde cada componente é associada a um cenário $k \in K$, denotando se a restrição de atraso para aquele cenário será ou não levada em conta no modelo (se será ou não relaxada). Assim, $z_k = 1$ representa que o caminho proposto poderá violar a restrição de atraso para o cenário k; caso contrário, o caminho deverá obrigatoriamente possuir atraso de percurso menor que D naquele cenário. Através dessas variáveis, limitamos os cenários violados sob o fator de risco α da forma descrita a seguir.

Denote por d_{uv}^k o valor do atraso do arco uv no cenário k, e seja ρ_k a probabilidade de uma amostra de ξ ser exatamente o cenário k dentro do conjunto finito K. Dessa forma, $d^k(u \rightsquigarrow v)$ denotará o atraso do caminho $u \rightsquigarrow v$ obtido no cenário k, definido como $d^k(u \rightsquigarrow v) := \sum_{ij \,\in\, u \rightsquigarrow v} d_{ij}^k$. Observe que

$$\sum_{k \in K} \rho_k = 1.$$

Assim, por **??**), podemos substituir a restrição (7c) pelas restrições lineares a seguir, onde $\mathcal{M}$ representa um valor positivo constante e suficientemente grande:

$$\sum_{uv \in A} d_{uv}^k x_{uv} \quad \leqslant \quad D + z_k \mathcal{M} \qquad \forall k \in K \tag{8a}$$

$$\sum_{k \in K} z_k \rho_k \quad \leqslant \quad \alpha \tag{8b}$$

$$z \in \{0,1\}^{|K|} \tag{8c}$$

Note que, nas restrições (8a), podemos escolher o valor de $\mathcal{M}$ de tal forma que, quando $z_k = 1$ para um dado cenário k, a restrição correspondente seja satisfeita por todos os caminhos de $s \Rrightarrow t$ em G, tornando-a redundante. Assim, nesse caso, o atraso de percurso da solução naquele cenário não é levado em consideração (isto é, a restrição é relaxada), podendo exceder o limite imposto. Entretanto, a restrição (8b) limita o número de variáveis z_k que podem assumir esse valor simultaneamente, fazendo com que a soma das probabilidades dos cenários cuja variável associada seja igual a 1 seja limitada superiormente por α. Dessa forma, as restrições (8) têm o mesmo papel na formulação que a restrição (7c). Temos, portanto, que a formulação linear (9) a seguir, que chamaremos de (P), é equivalente à versão não-determinística anteriormente apresentada:

$$(P) \qquad \min \quad \sum_{uv \in A} c_{uv} x_{uv} \tag{9a}$$

$$s.a. \quad \sum_{i \mid iv \in A} x_{iv} - \sum_{j \mid vj \in A} x_{vj} = \begin{cases} 1 & \text{se } v = s \\ -1 & \text{se } v = t \\ 0 & \text{caso contrário} \end{cases} \qquad \forall\, v \in N \tag{9b}$$

$$\sum_{uv \in A} d^k_{uv} x_{uv} \leqslant D + z_k \mathcal{M} \qquad \forall\, k \in K \tag{9c}$$

$$\sum_{k \in K} z_k \rho_k \leqslant \alpha \tag{9d}$$

$$x \in \{0,1\}^{|A|} \tag{9e}$$

$$z \in \{0,1\}^{|K|} \tag{9f}$$

3.4 Formulação Matricial

No Capítulo 4 descreveremos uma estratégia de decomposição em L para a resolução do problema. Exibimos a seguir a formulação obtida para o problema em sua representação matricial, de forma a facilitar a compreensão e o desenvolvimento da decomposição.

Seja $\mathcal{N}_{|V| \times |A|}$ a matriz de incidência nó-arco de G. Defina a matriz $\mathcal{E}_{|K| \times |A|}$ de forma que cada componente é exatamente o atraso de um dado arco em um dado cenário.

Defina o vetor-linha $\boldsymbol{\rho} := (\rho_1\ \rho_2\ \ldots\ \rho_{|K|})$, bem como os vetores-coluna $\mathbf{1} := (1\ \ 1\ \ \ldots\ \ 1)^t$, $\mathbf{0} := (0\ \ 0\ \ \ldots\ \ 0)^t$, $\mathbf{b}$, indexado pelos nós de G, tal que $\mathbf{b}_s = 1$, $\mathbf{b}_t = -1$ e $\mathbf{b}_i = 0\ \ \forall i \in N \setminus \{s,t\}$ e $\mathbf{c} \in \mathbb{R}^{|A|}_+$, indexado pelos arcos de G, onde cada componente é o custo do arco correspondente.

Com isso, reescrevemos (9) em sua versão matricial, que chamaremos de (P^M):

$$(P^M) \quad \min_{\substack{x \in \{0,1\}^{|A|} \\ z \in \{0,1\}^{|K|}}} \quad \mathbf{c}^t x + \mathbf{0}^t z \tag{10a}$$

$$s.a. \quad \mathcal{N} x = \mathbf{b} \tag{10b}$$

$$\mathcal{E} x \leqslant D\mathbf{1} + \mathcal{M} z \tag{10c}$$

$$\boldsymbol{\rho} z \leqslant \alpha \tag{10d}$$

3.5 Complexidade do Problema

Perceba que, para uma fixação qualquer de variáveis z em (P) de forma que a restrição (10d) seja satisfeita, obtemos uma instância do problema CMRL onde os recursos são dados pelos cenários associados às componentes nulas de z e os limites de consumo desses recursos são todos iguais e dados por D. As demais restrições podem ser descartadas, pois

como descrevemos, $\mathcal{M}$ é escolhido de forma que, para $z_k = 1$ com um dado cenário $k \in K$, todos os caminhos em $s \Rrightarrow t$ satisfaçam a restrição associada a k. Com isso, evidenciamos a relação entre os problemas CMRPAM e CMPL, o que nos leva a crer que tratamos com um problema difícil, dada a dificuldade do segundo problema, como afirmamos na Seção 2.2.2. Entretanto, não conseguimos encontrar demonstração formal para esse fato.

4 DECOMPOSIÇÃO EM L

Como abordagem inicial ao problema, buscamos resolvê-lo através de um particionamento com relação aos conjuntos de cenários cujas restrições podem ser relaxadas de forma a não ultrapassarmos a margem de risco α. Esses conjuntos são representados por vetores $\bar{z}$ que satisfazem a restrição (10d). Iterativamente, selecionamos $\bar{z}^i$, de uma maneira "esperta", como um desses vetores e determinamos o problema $(P)_{\bar{z}^i}$, obtido de (P) através das fixações das variáveis $z = \bar{z}^i$. Podemos perceber facilmente que soluções de $(P)_{\bar{z}^i}$ são viáveis para o problema (P), dada a forma como escolhemos $\bar{z}^i$.

Neste capítulo, exibimos um esquema de decomposição em L, no modelo mestre-escravo de Benders, que é responsável pela escolha "esperta" dos conjuntos de cenários que desejamos, bem como uma técnica de relaxação lagrangeana para obtenção de limites primais, desenvolvidos como tentativa inicial de abordagem ao problema. No Capítulo 5 descrevemos uma estratégia de enumeração desses vetores, de forma a construirmos as partições do problema (P).

4.1 Problemas mestre e escravos

Seja (P^M) a formulação matricial apresentada anteriormente para o problema do *Caminho Mínimo com Restrição Probabilística de Atraso Máximo*. Considere, para um dado $\overline{z}$ que satisfaça (10d), o problema $(P)_{\overline{z}}$, obtido através de (P^M) pela fixação das variáveis $z = \overline{z}$:

$$(P)_{\overline{z}} \qquad \min_{x \in \{0,1\}^{|A|}} \quad \mathbf{c}^t x \tag{11a}$$

$$s.a. \quad \mathcal{N}x = \mathbf{b} \tag{11b}$$

$$\mathcal{E}x \leqslant D\mathbf{1} + \mathcal{M}\overline{z} \tag{11c}$$

Associando as variáveis duais u às restrições (11b) e v às restrições (11c), obtemos o problema $(D)_{\overline{z}}$, dual da relaxação linear de $(P)_{\overline{z}}$, considerando que os limites $x \leqslant \mathbf{1}$ são implicados pela própria formulação de (P), dado que nenhum arco das soluções chega em s ou sai de t.

$$(D)_{\overline{z}} \quad \max_{u,v} \quad u\mathbf{b} + v(D\mathbf{1} + \mathcal{M}\overline{z}) \tag{12a}$$

$$s.a. \quad u\mathcal{N} + v\mathcal{E} \leqslant \mathbf{c}^t \tag{12b}$$

$$u \in \mathbb{R}^{|A|}, \quad v \in \mathbb{R}_-^{|K|} \tag{12c}$$

Observe que $(D)_{\overline{z}}$, definido sobre variáveis contínuas, e $(P)_{\overline{z}}$, definido sobre variáveis inteiras, formam um par primal-dual fraco. Ainda assim, podemos utilizar soluções de $(D)_{\overline{z}}$ para obter cortes de viabilidade ou de otimalidade para (P).

4.1.1 Cortes de viabilidade e otimalidade

Proposição 1. *Seja* $(\overline{u}, \overline{v})$*, para um dado* $\overline{z}$ *satisfazendo* (10d)*, uma solução ótima e limitada de* $(D)_{\overline{z}}$*, de valor denotado por* $\overline{w} = \overline{u}\mathbf{b} + \overline{v}(D\mathbf{1} + \mathcal{M}\overline{z})$*, e seja* $(\hat{u}, \hat{v})$ *uma solução ilimitada (raio extremo) de* $(D)_{\hat{z}}$*, associado a* $\hat{z}$ *que também satisfaz* (10d)*. Denote por* (u, v) *a solução ótima, de valor* w*, de* $(D)_z$ *para um* z *genérico igualmente satisfazendo* (10d)*. Então, um corte de otimalidade para* (P) *é dado por*

$$w \geqslant \overline{w} + \overline{v}\mathcal{M}(z - \overline{z}) \tag{13}$$

e um corte de viabilidade para (P) *é dado por*

$$0 \geqslant \hat{u}\mathbf{b} + \hat{v}(D\mathbf{1} + \mathcal{M}z) \tag{14}$$

Demonstração. Como (u, v) é uma solução ótima para $(D)_z$, por otimalidade, $w = u\mathbf{b} + v(\mathbf{1}D + \mathcal{M}z) \geqslant \overline{u}\mathbf{b} + \overline{v}(D\mathbf{1} + \mathcal{M}z)$. Portanto,

$$\begin{aligned} w &\geqslant \bar{u}\mathbf{b} + \bar{v}(D\mathbf{1} + \mathcal{M}z) \\ &= \bar{u}\mathbf{b} + \bar{v}(D\mathbf{1} + \mathcal{M}z + \mathcal{M}\bar{z} - \mathcal{M}\bar{z}) \\ &= \bar{u}\mathbf{b} + \bar{v}(D\mathbf{1} + \mathcal{M}\bar{z}) + \bar{v}\mathcal{M}z - \bar{v}\mathcal{M}\bar{z} \\ \therefore \quad w &\geqslant \bar{w} + \bar{v}\mathcal{M}(z - \bar{z}) \end{aligned}$$

Sejam agora (u^s, v^s) e $(\hat{u}, \hat{v})$ respectivamente uma direção de recessão e um ponto extremo de $(D)_{\hat{z}}$. Seja $\lambda > 0$ uma constante não-negativa e considere todos os pontos do raio $(\hat{u}, \hat{v}) + \lambda(u^s, v^s)$. Caso $u^s\mathbf{b} + v^s(\mathbf{1}D + \mathcal{M}\hat{z}) > 0$, então para $\lambda \to \infty$ a solução de $(D)_{\hat{z}}$ torna-se ilimitada nos pontos ao longo do raio. Portanto, um corte de viabilidade será dado por

$$0 \geqslant \hat{u}\mathbf{b} + \hat{v}(D\mathbf{1} + \mathcal{M}z)$$

□

4.1.2 Esquema de decomposição

Utilizando os cortes demonstrados acima e considerando *Vertices*$((D)_z)$ o conjunto de vértices do problema $(D)_z$, (onde cada $(\overline{u}, \overline{v})$ é a solução ótima do problema $(D)_{\bar{z}}$ associado a um respectivo $\bar{z}$, representados por $(\overline{u}, \overline{v} \mid \overline{z}) \in$ *Vertices*$((D)_z)$), e *Raios*$((D)_z)$ o conjunto de raios extremos de $(D)_z$, obtemos o seguinte esquema de decomposição em L para (P).

$$(M) \quad \min_{\substack{z \in \{0,1\}^{|K|} \\ w \in \mathbb{R}}} w \tag{15a}$$

$$s.a. \quad w \geqslant \bar{w} + \bar{v}\mathcal{M}(z - \bar{z}) \qquad \forall\, (\overline{u}, \overline{v} \mid \overline{z}) \in \textit{Vertices}((D)_z) \tag{15b}$$

$$0 \geqslant \hat{u}\mathbf{b} + \hat{v}(D\mathbf{1} + \mathcal{M}z) \qquad \forall\, (\hat{u}, \hat{v}) \in \textit{Raios}((D)_z) \tag{15c}$$

$$\boldsymbol{\rho} z \leqslant \alpha \tag{15d}$$

onde (M) é o problema mestre e $(D)_{\bar{z}}$ são os problemas escravos, para cada $\bar{z}$ solução de (M). Como se trata de um problema mestre de Benders obtido a partir de (P), o valor da solução ótima de (M) é um limite inferior para o custo da solução de (P). Apenas um subconjunto dos cortes presentes na formulação de (M) é determinante e necessário para obtermos esse limite. Entretanto, não sabemos *a priori* quais desses cortes pertencem a tal subconjunto.

Ainda que consigamos enumerar com (M) todas as soluções viáveis z de (P), pode ser que o valor ótimo de (M) ainda seja inferior ao ótimo de (P). Entretanto, o que nos interessa é enumerar de uma maneira inteligente essas soluções z.

A cada iteração i, resolvemos uma relaxação de (M), denominada $(M)^i$, onde apenas um subconjunto dos cortes (15b)–(15c) de (M) são considerados, obtendo uma solução $\bar{z}^{i+1}$. Com essa solução, determinamos a adição de um corte de otimalidade ou de viabilidade de (M) a $(M)^i$, através da resolução do problema $(D)_{\bar{z}^i}$ associado, de forma a obtermos o problema $(M)^{i+1}$ a ser resolvido na próxima iteração. Note que soluções dos problemas $(M)^i$, $\forall i$, por conta da restrição (15d), representam conjuntos de cenários tais que a relaxação das suas restrições de atraso correspondentes em (P) preserva o limite de risco desejado. Dessa forma, temos que as soluções dos problemas $(P)_{\bar{z}^i}$ são soluções viáveis para (P).

Portanto, se possuíssemos todos os vértices dos problemas $(D)_z$, a família dos problemas $(P)_{\bar{z}^i}$ representaria um particionamento de (P), onde a solução ótima para este reside na solução de menor custo dentre as das partições.

4.1.3 Escolha do vetor inicial

Como ponto de partida para o esquema de decomposição apresentado, podemos tomar qualquer vetor $\overline{z}$ que satisfaça (10d). Entretanto, decidimos partir de um vetor maximal com relação à quantidade de componentes de valor 1, escolhido segundo o Algoritmo 1.

Supomos que a função INDICESND($\boldsymbol{\rho}$) retorna um vetor de índices de $\boldsymbol{\rho}$ tal que, se percorrermos este último na ordem de índices especificada pelo retorno da função, estaremos obtendo os valores em $\boldsymbol{\rho}$ do menor para o maior.

Na linha 2 do Algoritmo 1, definimos o vetor auxiliar $\overline{z}$ como um vetor nulo de dimensão $|K|$. No laço encontrado na linha 4, percorremos o vetor $\boldsymbol{\rho}$ na ordem obtida na linha 3 verificando se, ao tornar 1 o índice correspondente no vetor $\overline{z}$, este deixaria de satisfazer a restrição (10d). Caso sim, retornamos o vetor obtido até o momento. Caso contrário, modificamos o valor desse índice no vetor a ser retornado.

Ao final da execução do algoritmo, temos um vetor maximal em componentes não-nulas e que satisfaz a restrição (10d).

4.2 Obtenção de limites superiores

Perceba que a obtenção de limites superiores com esse esquema de decomposição não é direta, dado que não lidamos com um par primal-dual forte. Assim sendo, propomos o

Algoritmo 1: Algoritmo de escolha do vetor inicial para o esquema de decomposição em L apresentado.

Chamada: ZINICIAL($\boldsymbol{\rho}, \alpha$)
Entrada: Vetor $\boldsymbol{\rho}$ de probabilidades dos cenários a considerar, fator de risco α
Resultado: Vetor $\overline{z} \in \{0,1\}^{|K|}$ maximal tal que $\boldsymbol{\rho}\overline{z} \leqslant \alpha$

```
1  ᾱ ← 0
2  z̄ ← 0_{|K|}
3  I ← INDICESND(ρ)
4  para i ← 1, ..., |I| faça
5      se ᾱ + ρ_i ⩽ α então
6          ᾱ ← ᾱ + ρ_i
7          z̄_i ← 1
8      senão
9          saia
10 retorne z̄
```

uso das soluções de $(D)_z$ para obter tais limites da forma como segue. Relaxamos o problema (P) associando multiplicadores de Lagrange $\bar{v}$ às restrições (10c) (variáveis duais cujos valores são obtidos pela solução de $(D)_{\bar{z}}$ para um dado $\bar{z}$), levando essas restrições à função objetivo do problema. Determinamos, assim, o problema $(\overline{P}_{\bar{v}})$ a seguir. Esse novo problema nada mais é que um problema de caminho mínimo com perturbações nos custos dos arcos. Note que a solução de $(\overline{P}_{\bar{v}})$ é um $\hat{x}$ inteiro, o que nos permite concluir que não necessariamente $\hat{x}$ seja a solução ótima dual de $(D)_{\bar{z}}$. Ainda assim, essa solução pode ser usada na obtenção de um limite superior para (P) de acordo com a Proposição 2.

Proposição 2. *Para um dado $\bar{v} \leqslant 0$, seja $\bar{x}$ uma solução ótima para o problema $(\overline{P}_{\bar{v}})$ a seguir:*

$$(\overline{P}_{\bar{v}}) \qquad \min \quad (\mathbf{c}^t - \bar{v}\mathcal{E})x \tag{16a}$$

$$s.a. \quad \mathcal{N}x = \mathbf{b} \tag{16b}$$

$$x \in [0;1]^{|A|} \tag{16c}$$

Defina o conjunto $L = \{\ell \mid \mathcal{E}(\ell)\bar{x} > D\}$, onde $\mathcal{E}(\ell)$ representa a linha ℓ de $\mathcal{E}$ e $\bar{x}$ representa a solução ótima do problema, e seja $\tilde{z}$ definido como segue:

$$\tilde{z}_\ell = \begin{cases} 1, & \textit{se } \ell \in L; \\ 0, & \textit{caso contrário.} \end{cases}$$

Se $\bar{x}$ é tal que $\sum_{\ell \in L} \boldsymbol{\rho}_\ell \leqslant \alpha$, então $(\bar{x}, \tilde{z})$ é uma solução viável para o problema (P). Portanto, $\mathbf{c}^t\bar{x}$ é um limite superior para a solução ótima de (P).

Demonstração. Perceba que, como $\bar{v} \leqslant 0$, os custos perturbados em $(\overline{P}_{\bar{v}})$ são todos positivos. Com isso, observe que $\bar{x}$ é o vetor característico de um caminho em G e que a soma das probabilidades dos cenários associados às componentes não-nulas do vetor $\tilde{z}$ (por definição, os cenários

nos quais o atraso de percurso de $\bar{x}$ ultrapassa o limite D) é menor ou igual a α. Portanto, $(\bar{x}, \tilde{z})$ é uma solução viável de (P). □

5 ESQUEMA DE ENUMERAÇÃO DE PARTIÇÕES DE (P)

Neste capítulo, apresentamos uma técnica de enumeração de partições para o problema (P) fazendo uso da decomposição em L descrita no Capítulo 4.

Essa enumeração é feita aplicando-se o algoritmo de *Branch-and-Bound* sobre o espaço das variáveis z de (M), obtendo-se soluções $\bar{z}$ com as quais determinamos uma nova partição $(P)_{\bar{z}}$ a explorar e um novo corte de otimalidade ou de viabilidade a ser adicionado globalmente a (M). Resolvemos a cada iteração b um problema mestre local $(M)^b$, consistindo da relaxação de (M) com todos os cortes já enumerados e acrescido das devidas fixações de variáveis. Para cada solução $\bar{z}^b$ de um dado nó na árvore de *B&B*, passamos a buscar pela solução ótima do problema auxiliar $(P)_{\bar{z}^b}$ associado, que nos fornecerá um limite superior para o problema, como já enunciamos. Obtemos, então, os valores das variáveis duais (soluções ótimas de $(D)_{\bar{z}^b}$) de forma a gerar os cortes necessários introduzidos na Seção 4.1.1, válidos globalmente para o problema mestre, de onde retiramos um novo limite inferior local para (P), que utilizamos na determinação de um limite global.

Nesse esquema, utilizamos uma variação da estratégia de ramificação recentemente introduzida em Freitas (2011) para problemas cuja relaxação é um problema de otimização combinatória em variáveis 0-1.

Observe que não poderíamos tentar obter z a partir de valores para x, dado que uma vez fixado um caminho, o conjunto de cenários para os quais esse caminho viola o limite máximo de tempo pode ser determinado trivialmente ao contabilizarmos seus tempos de percurso em cada cenário. Essa forte relação nos impede de obter informações úteis sobre as soluções por esse tipo de abordagem.

Nas subseções a seguir, descrevemos as ferramentas auxiliares à técnica desenvolvida. Discutimos também uma desigualdade válida globalmente que propomos para evitar percorrer nós da árvore de *B&B* que certamente não levariam a uma melhora das soluções já encontradas ao longo da avaliação dos nós. No Capítulo 8 são exibidos os resultados computacionais referentes à implementação desse esquema.

5.1 Desigualdade de subconjuntos

Na Seção 3.3.2, apontamos o fato de que, quando $z_k = 1$ para algum $k \in K$, podemos desprezar a restrição de atraso máximo relacionada ao cenário k em (P), dado que ela será satisfeita por qualquer caminho de $s \Rrightarrow t$ em G. Note que fixar uma variável $z_k = 1$ não implica que o atraso da solução obtida irá obrigatoriamente ultrapassar o limite D desejado, mas sim que não estamos interessados nesse atraso para aquele cenário. Podemos interpretar essa fixação como uma "permissão de violação" da restrição de atraso máximo.

5.1.1 Dominância entre vetores

Defina a relação "$\leqslant$" entre vetores binários $\mathbf{a}$ e $\mathbf{b}$ de mesma dimensão de forma que $\mathbf{a} \leqslant \mathbf{b}$ se, e somente se, $\forall i \ \ \mathbf{a}_i \leqslant \mathbf{b}_i$. Se $\mathbf{a}$ e $\mathbf{b}$ são tais que $\mathbf{a} \nleqslant \mathbf{b}$ e $\mathbf{b} \nleqslant \mathbf{a}$, esse par de vetores é dito *incomparável*.

Sejam $\hat{z}$ e $\bar{z}$ vetores que satisfazem a restrição (10d) tais que $\hat{z} \leqslant \bar{z}$. Tome $\hat{x}$ e $\bar{x}$ soluções ótimas de $(P)_{\hat{z}}$ e $(P)_{\bar{z}}$, obtidos de (P) respectivamente pela fixação das variáveis z em $z = \hat{z}$ e $z = \bar{z}$.

Veja que, pela definição da relação "$\leqslant$", segue que ou $\hat{z} = \bar{z}$, ou existe ao menos um i tal que $\hat{z}_i = 0$ e $\bar{z}_i = 1$. Portanto, podemos concluir que ou $\mathbf{c}^t\hat{x} = \mathbf{c}^t\bar{x}$ devido ao primeiro caso, ou $\mathbf{c}^t\hat{x} \geqslant \mathbf{c}^t\bar{x}$, dado que $\bar{z}_i = 1$ no segundo caso permite às soluções viáveis de $(P)_{\bar{z}}$ a violação do limite de atraso máximo no cenário i, aumentando assim o espaço de soluções deste problema com relação a $(P)_{\hat{z}}$.

Dizemos que as soluções de $(P)_{\bar{z}}$ *dominam* as soluções de $(P)_{\hat{z}}$ ou, de uma forma mais simples, que $\bar{z}$ *domina* $\hat{z}$. Pelas afirmações anteriores, concluímos que não vale a pena buscar por soluções associadas a um dado z' no esquema *B&B* quando já se conhece uma solução ótima associada a qualquer z'' que o domina.

Evitamos essa situação no nosso esquema através da adição do que chamamos de *corte de subconjuntos relativo a* z' ao problema mestre (M), que é responsável por separar do espaço de soluções de (M) todos os vetores dominados por z'.

5.1.2 Corte de subconjuntos

O seguinte resultado facilita a compreensão da validade do corte de subconjuntos apresentado nesta seção.

Teorema 1. *Sejam* $\mathbf{a}$ *e* $\mathbf{b}$ *vetores binários.* $\mathbf{a} \leqslant \mathbf{b}$ *se, e somente se,* $\mathbf{a}$ *não difere de* $\mathbf{b}$ *nas componentes* i *onde* $\mathbf{b}_i = 0$.

Demonstração.

($\Rightarrow$) Sejam $\mathbf{a}$ e $\mathbf{b}$ tais que $\mathbf{a} \leqslant \mathbf{b}$ e tome um i qualquer tal que $\mathbf{b}_i = 0$. $\mathbf{a}_i$ deve ser obrigatoriamente 0, pois caso contrário, pela definição, encontraríamos um absurdo em relação à hipótese de que $\mathbf{a} \leqslant \mathbf{b}$. Portanto, se $\mathbf{a} \neq \mathbf{b}$, essa diferença ocorre em alguma componente i onde $\mathbf{b}_i = 1$.

($\Leftarrow$) Sejam $\mathbf{a}$ e $\mathbf{b}$ tais que $\forall i$ onde $\mathbf{b}_i = 0$, temos $\mathbf{a}_i = 0$. Assim, para essas componentes, vale a relação $\mathbf{a}_i \leqslant \mathbf{b}_i$. Nas demais componentes j, temos que $\mathbf{b}_j = 1$. Como $\mathbf{a}$ e $\mathbf{b}$ são vetores binários, segue que $\forall j \ \ \mathbf{a}_j \leqslant \mathbf{b}_j$, concluindo a prova.

□

Pelo que afirmamos na Seção 5.1.1 e pelo Teorema 1, temos que não nos interessam os vetores que preservam todas as componentes não-nulas de um dado vetor z' encontrado

anteriormente no esquema *B&B*, com isso desprezando todos os vetores dominados por z'. Assim, obtemos o corte de subconjuntos desejado.

Corolário 2. *A desigualdade a seguir separa do espaço de soluções viáveis todos os vetores dominados por um dado vetor* z'*:*

$$\sum_{i \mid z'_i = 0} z_i \geqslant 1 \tag{17}$$

Demonstração. Segue diretamente do Teorema 1. □

5.2 Ramificação

A cada iteração b do algoritmo de *B&B*, devemos resolver o problema local associado ao nó n^b atual e, a partir da solução obtida, ramificar novos nós da árvore. Chamaremos os nós ramificados a partir de um nó n^b de *descendentes* desse nó.

A solução obtida em cada nó n^b é uma solução ótima de um problema $(M)^b$ que possui apenas um subconjunto dos cortes de (M), além de algumas fixações de variáveis, e trata-se, portanto, de um vetor $\bar{z}^b \in \{0;1\}^{|K|}$ ótimo localmente (na sub-árvore enraizada em n^b) que satisfaz a restrição (10d). Desejamos ramificar n^b de forma que a sua solução não esteja contida no espaço viável de qualquer de seus descendentes e que esses espaços formem uma partição do espaço viável de n^b excluindo-se a solução $\bar{z}^b$. Para isso utilizamos um esquema de ramificação baseado no trabalho de Freitas (2011), com algumas adequações ao nosso problema.

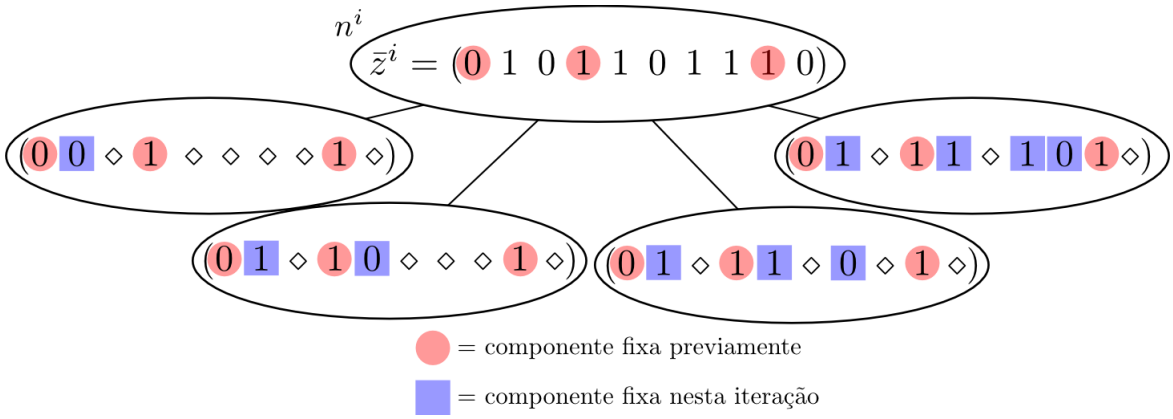
Podemos visualizar a ideia de Freitas (2011) da seguinte forma: para um dado vetor solução $\bar{z}^b$, seja $\mathsf{um}(\bar{z}^b)$ o conjunto de índices de $\bar{z}^b$ tais que os valores correspondentes em $\bar{z}^b$ valem 1. Isso significa que na solução do problema $(P)_{\bar{z}^b}$ permitimos a excedência do valor de atraso para aqueles cenários. Dentre esses índices, tome aqueles cuja variável z correspondente em n^b não foi fixada e denote esse novo conjunto por $\mathcal{L}(\bar{z}^b)$. A partir desse conjunto ramificaremos os descendentes de n^b.

Para cada índice i em $\mathcal{L}(\bar{z}^b)$ geramos um novo nó onde analisamos o problema local associado a n^b acrescido da fixação em 0 da variável z_i correspondente. Isso garante que a solução obtida em n^b não pertença ao espaço de nenhum de seus descendentes. Entretanto, isso não é suficiente para garantir que os espaços dos descendentes sejam disjuntos entre si. Basta tomarmos um vetor $\hat{z}$ tal que, para $i, j \in \mathcal{L}(\bar{z}^b)$ distintos, $\hat{z}_i = \hat{z}_j = 0$ e $\hat{z}_k = \bar{z}^b_k$ nas demais componentes k. Perceba que $\hat{z}$ seria uma solução viável para ambos os descendentes associados a i e j. Para eliminar esse caso, sendo $\preccurlyeq$ uma ordem total sobre os elementos de $\mathcal{L}(\bar{z}^b)$, acrescentamos a fixação em 1 das variáveis z_j, $\forall j \in \mathcal{L}(\bar{z}^b) \setminus \{i\}$ tal que $j \preccurlyeq i$, supondo que i é o índice associado ao descendente em questão.

Assim, as soluções viáveis de um nó n^b_i, descendente de n^b e associado ao índice i, diferem da de n^b na componente i. Além disso, x_i será fixa em 1 em todos os nós ramificados após n^b_i, fazendo com que seus espaços de soluções sejam disjuntos. Dessa forma, cada

nó descendente verifica o caso em que o atraso da solução deve obrigatoriamente satisfazer o limite máximo no cenário relacionado, em contrapartida à solução de n^b. A Figura 8 ilustra um exemplo de ramificação realizada por essa técnica.

Figura 8 – Exemplo de ramificação realizada pela técnica de Freitas (2011)



Fonte: própria autoria

O nó marcado como n^i foi o nó escolhido para ramificação. A solução $\bar{z}^i$ de n^i é exibido. Os novos nós criados e seus valores fixos são exibidos abaixo de n^i. O símbolo $\diamond$ substitui os valores desconhecidos das variáveis nos nós ainda não avaliados. Neste exemplo, supomos que $i \preccurlyeq j \iff i \leqslant j$.

Aproveitando-nos do conceito de *dominância entre vetores* que introduzimos na Seção 5.1.1, decidimos acrescentar à estratégia em Freitas (2011) uma forma de considerar também vetores que venham a dominar a solução do nó corrente. Antes de apresentarmos nossa estratégia, será necessária a introdução de uma nova definição.

Denotando por $\mathsf{zero}(\bar{z})$ o conjunto de índices das componentes de valor nulo do vetor binário $\bar{z}$, dizemos que um vetor $\bar{z}$ é *expansível com relação a* α (ou apenas *expansível*) quando existe ao menos um $i \in \mathsf{zero}(\bar{z})$ tal que $\boldsymbol{\rho}\bar{z} + \rho_i \leqslant \alpha$. Dizemos, nesse caso, que o índice i *expande* $\bar{z}$. Isso ilustra o fato de que, caso o vetor solução $\bar{z}^b$ em um nó n^b seja expansível com relação a α, ainda existem vetores que satisfazem (10d) que o dominam, nos permitindo decidir por abandonar o nó atual e passar a considerar alguns desses vetores. Assim, caso a solução $\bar{z}^b$ seja expansível preservando as componentes fixas, ramificaremos o nó n^b observando não $\bar{z}^b$ mas sim um vetor maximal que o domine.

A seguir, mostramos um resultado que nos permite determinar quais partições do problema (P) são suficientes serem analisadas de forma a explorarmos todo o espaço de soluções onde temos a possibilidade de encontrar a solução ótima.

Teorema 3. *A solução ótima de* (P) *reside em uma das partições* $(P)_{\hat{z}}$*, onde* $\hat{z}$ *é um vetor maximal da ordem* $\leqslant$*, definida dentre os que satisfazem a restrição* (15d).

Demonstração. Suponha que $(\tilde{x}, \tilde{z})$ seja uma solução ótima para (P) e assuma que $\tilde{z}$ seja expansível. Pelo conceito de dominância, qualquer partição $(P)_{\hat{z}}$, onde $\hat{z}$ é um vetor arbitrário

que domina $\tilde{z}$, tem $\tilde{x}$ como um caminho viável. Em particular, se tomarmos $\hat{z}$ maximal quanto à ordem $\leqslant$ de forma que $\hat{z}$ satisfaz a restrição (15d), temos que $\tilde{x}$ é a solução ótima dessa partição, o que nos permite concluir o resultado. □

Com isso podemos concluir que, se preservarmos todos os vetores maximais que pertencem ao espaço de soluções do nó n^b ao modificarmos sua ramificação, estaremos garantindo a obtenção da solução ótima de (P), quando existir. O Teorema 4 demonstra que, realizando a ramificação de n^b visualizando um dos vetores maximais que dominam sua solução $\bar{z}^b$, garantimos essa preservação.

Teorema 4. *Seja n^b um nó da árvore de* B&B*, com $\bar{z}^b$ a solução ótima do seu problema associado. Seja $\hat{z}^b$ maximal tal que $\bar{z}^b \leqslant \hat{z}^b$, preservando-se as componentes cujas variáveis correspondentes estejam fixas. Considere o conjunto $\mathfrak{D}(\hat{z}^b)$ como sendo o conjunto de nós obtidos ao ramificarmos $\hat{z}^b$ segundo a estratégia de Freitas (2011). Temos que a união dos espaços de soluções dos nós em $\mathfrak{D}(\hat{z}^b)$ contém todos os vetores maximais que pertencem ao espaço de soluções de n^b.*

Demonstração. Sejam $i_1, i_2, \ldots, i_j$ os índices nos quais $\hat{z}^b$ difere de $\bar{z}^b$ de forma que $i_1 \preccurlyeq i_2 \preccurlyeq \ldots \preccurlyeq i_j$. Como $\hat{z}^b$ é maximal, devemos mostrar que nos espaços de $\mathfrak{D}(\hat{z}^b)$ constam os demais vetores maximais desejados que sejam diferentes de $\hat{z}^b$.

Assim, esses vetores devem diferir de $\hat{z}^b$ em ao menos uma das componentes i, dado que um vetor que preserve essas componentes deve necessariamente ser igual a $\hat{z}^b$, por este ser maximal. Seja $\tilde{z}^b$ um desses vetores maximais, com $\tilde{z}^b \neq \hat{z}^b$, e seja i_ℓ o primeiro dos índices em que $\tilde{z}^b$ difere de $\hat{z}^b$, segundo a ordem total $\preccurlyeq$.

Pela estratégia de Freitas (2011), existe em $\mathfrak{D}(\hat{z}^b)$ um nó em que z_{i_ℓ} está fixo em 0, as componentes anteriores estão fixas em 1 e as demais estão livres, segundo $\preccurlyeq$. Como i_ℓ é a primeira componente em que ocorre essas diferença, temos que os valores das demais podem ser obtidos através desse nó, o que mostra que $\tilde{z}^b \in \mathfrak{D}(\hat{z}^b)$.

Pela generalidade nas escolhas dos vetores, temos que $\mathfrak{D}(\hat{z}^b)$ contém todos os vetores maximais contidos nos espaços dos nós em $\mathfrak{D}(\hat{z}^b)$, o que é suficiente para demonstrar que a mudança nas profundidades dos nós na árvore de *B&B* gerada pela mudança na ramificação de n^b preserva os ramos onde poderemos encontrar a solução ótima para (P). □

Para garantirmos que não vamos repetir a solução do nó-pai n^b, após sua ramificação, procedemos à adição do corte de subconjuntos relacionado ao vetor solução $\hat{z}^b$, válido globalmente.

5.3 Limites globais

Consideramos, no nosso esquema de enumeração, dois tipos distintos de nós, que denotaremos por $\mathfrak{m}$ e $\mathfrak{p}$. Nós $\mathfrak{m}$ são associados a problemas locais (M) e são responsáveis pela enumeração das partições do problema (P), dadas por suas soluções $\bar{z}$. Esses nós são

ramificados em um conjunto de novos nós do tipo $\mathfrak{m}$, usando-se a estratégia de ramificação descrita, e em um novo nó do tipo $\mathfrak{p}$. Nós do tipo $\mathfrak{p}$ são associados às partições $(P)_{\bar{z}}$ dadas pela solução do nó do tipo $\mathfrak{m}$ que lhe deu origem, e são responsáveis por buscar soluções ótimas dessas partições. Esses nós são ramificados em dois outros novos nós do tipo $\mathfrak{p}$, da forma como descrevemos mais à frente.

A menor das cotas associadas a nós $\mathfrak{m}$ ativos na árvore de *B&B* é também uma cota inferior para (P), dados que esses nós são associados a relaxações do problema mestre (M). Dizemos que um nó está *ativo* quando ele já foi alcançado pelo algoritmo de busca do *B&B* mas ainda não foi avaliado. Com relação a nós do tipo $\mathfrak{p}$, temos que esses são associados a versões restritas de (P), onde fixamos algumas de suas variáveis, mas preservamos as demais restrições do problema original. Portanto, pela forma como escolhemos as fixações, soluções inteiras obtidas nesse tipo de nó fornecem cotas superiores para (P), por este tratar-se de um problema de minimização.

Durante a busca de soluções ótimas em nós $\mathfrak{p}$, trabalhamos com a relaxação linear do problema associado e continuamos o processo realizando a ramificação desses nós por *dicotomia do mais fracionário*, ou seja, escolhemos uma variável x_i cuja solução no nó atual mais se aproxima de $\frac{1}{2}$ e consideramos dois novos nós $\mathfrak{p}$, onde tal variável é fixada respectivamente em $x_i = 0$ e $x_i = 1$.

5.3.1 Atualização dos limites

Escolhemos o nó com melhor valor de solução da lista de nós ativos (a árvore de *B&B* é representada como um *heap*) e resolvemos seu problema associado. Caso trate-se de um nó $\mathfrak{m}$, determinamos o mínimo entre o valor da sua solução ótima e o menor valor das soluções encontradas nos nós-pai daqueles na lista de ativos. Caso esse valor melhore a nossa cota inferior global, temos um novo limite inferior. No caso de escolhermos um nó $\mathfrak{p}$, observamos se a solução encontrada é inteira. Nesse caso, verificamos se o seu valor melhora a nossa cota superior e a atualizamos se conveniente.

5.3.2 Poda dos nós

Sejam LB e UB os valores das melhores cotas inferior e superior já obtidas. Suponha que obtivemos da lista de nós ativos um nó $\mathfrak{m}$ cujo valor de solução seja $\bar{w}$. Como $\bar{w}$ é um limite inferior local à sub-árvore *B&B* enraizada neste nó, caso $\bar{w} > UB$ podemos concluir que não obteremos soluções melhores que as já encontradas se ramificarmos aquele nó.

Suponha agora tratar-se de um nó $\mathfrak{p}$ cujo valor da solução seja $\mathbf{c}\bar{x}$. Note que estes nós fornecem cotas inferiores para o valor da solução do problema associado $(P)_{\bar{z}}$ para um dado $\bar{z}$, que por sua vez nos dá um limite superior para (P). Assim, caso $\mathbf{c}\bar{x} > UB$, não obteremos um melhor limite superior que aquele que já dispomos se continuarmos a ramificar este nó. Além disso, supondo que a solução do nó em questão seja inteira, $\mathbf{c}\bar{x}$ já é uma cota superior para (P). Por isso, em todo caso, não se faz mais necessária a ramificação.

Outro processo de poda trivialmente realizado se dá quando o problema associado ao nó escolhido for inviável.

5.4 Pré-fase

De forma a acelerar o processo de solução do problema mestre da decomposição que apresentamos, usamos um artifício bastante praticado em trabalhos que lidam com decomposições de problemas inteiros. Note que os cortes apresentados na Seção 4.1.1 são válidos para qualquer z que satisfaça (10d), mas não estão restritos a vetores inteiros. Assim, em uma pré-fase ao algoritmo, podemos resolver a relaxação linear do problema mestre (M), que denotaremos por $(\overline{M})$, usando o mesmo esquema de cortes e, ao encontrarmos sua solução ótima ou quando o critério de parada for satisfeito, podemos reaproveitar todos os cortes adicionados a $(\overline{M})$ em (M) como forma de obtermos uma "partida a quente" na resolução deste último.

6 MELHORA DAS COTAS INFERIORES

Neste capítulo descrevemos a estratégia desenvolvida de forma a tentar obter uma melhora nos limites inferiores obtidos pelo esquema de enumeração descrito no Capítulo 5. Nessa estratégia, a cada nó da árvore de *B&B* verificamos a solução de uma nova relaxação do problema (P) baseada num resultado que relaciona os dois problemas, resolvida através do uso de penalizações Lagrangeanas cujos multiplicadores são obtidos através do método do subgradiente. Nas próximas seções descrevemos a obtenção do novo problema e demonstramos o resultado citado, além de detalharmos o método utilizado de forma a obtermos as novas cotas inferiores desejadas. No Capítulo 8, reportamos os resultados computacionais para o esquema de enumeração de partições ao adicionarmos a técnica descrita.

6.1 Relaxação

Inicialmente, exibimos a demonstração para a Proposição 3 a seguir, que nos auxilia na construção dos novos limites inferiores dos nós do *B&B*.

Proposição 3. *Sejam (Q) e $(\overline{Q})$ os problemas a seguir*

$$
\begin{array}{llll}
(Q) & \min & \mathbf{c}^t x & \\
 & s.a. & Ax & \leqslant \mathbf{b} \\
 & & x \in \{0,1\}^p &
\end{array}
\qquad\qquad
\begin{array}{llll}
(\overline{Q}) & \min & \mathbf{c}^t x & \\
 & s.a. & \boldsymbol{\delta}^t Ax & \leqslant \boldsymbol{\delta}^t \mathbf{b} \\
 & & x \in \{0,1\}^p &
\end{array}
$$

onde $\boldsymbol{\delta} \in \mathbb{R}^q_+$. Temos que o valor da solução ótima de $(\overline{Q})$ é um limite inferior para o valor da solução de (Q).

Demonstração. Por se tratarem de dois problemas de minimização, temos que para demonstrar a afirmação basta mostrarmos que uma solução ótima para (Q) é uma solução viável para $(\overline{Q})$, com $\beta \geqslant 0$.

Sejam a^i os vetores-linha de A, com $1 \leqslant i \leqslant q$, e seja $x^\star$ uma solução ótima para (Q). Assim,

$$
\begin{array}{rcl}
a^1 x^\star \leqslant b_1 & \Longrightarrow & \delta_1 a^1 x^\star \leqslant \delta_1 b_1 \\
a^2 x^\star \leqslant b_2 & \Longrightarrow & \delta_2 a^2 x^\star \leqslant \delta_2 b_2 \\
\vdots & & \vdots \\
a^q x^\star \leqslant b_q & \Longrightarrow & \delta_q a^q x^\star \leqslant \delta_q b_q \quad (+) \\
 & & \hline \\
 & & \displaystyle\sum_{1 \leqslant i \leqslant q} \delta_i a^i x^\star \leqslant \sum_{1 \leqslant i \leqslant q} \delta_i b_i \Longrightarrow \quad \delta^t A x^\star \leqslant \delta^t b
\end{array}
$$

e, portanto, $x^\star$ é uma solução viável de $(\overline{Q})$. □

Assim, seja n^b o nó da árvore de *B&B* analisado na b-ésima iteração do algoritmo e seja $\bar{z}^b$ a solução ótima para o problema $(M)^b$ associado. Defina $\bar{\boldsymbol{\rho}}^b$ de forma que

$$\bar{\boldsymbol{\rho}}_i^b = \begin{cases} \rho_i & \text{, se } \bar{z}_i^b = 0 \\ 0 & \text{, c.c} \end{cases}$$

ou seja, as componentes de $\bar{\boldsymbol{\rho}}^b$ para as quais decidimos respeitar o limite de atraso no cenário associado têm valor igual às probabilidades desses cenários, as demais tendo valor nulo.

Definimos o problema $(S)^b$ a seguir:

$$\begin{aligned} (S)^b \qquad \min \quad & \mathbf{c}^t x && && (18a) \\ s.a. \quad & \mathcal{N}x && = \mathbf{b} && (18b) \\ & \bar{\boldsymbol{\rho}}^b \mathcal{E} x && \leqslant \bar{\boldsymbol{\rho}}^b D\mathbf{1} && (18c) \\ & x \in [0,1]^m && && (18d) \end{aligned}$$

Perceba que a restrição (18c) é uma combinação linear das restrições de atraso de (P) ponderadas pelas probabilidades dos cenários para os quais devemos respeitar o limite de atraso. Segundo a Proposição 3, soluções para $(S)^b$ fornecem cotas inferiores para as soluções de $(\overline{P})_{\bar{z}^b}$.

De forma a resolvermos esse novo problema, relaxamos a restrição (18c), levando-a à função objetivo e introduzindo um penalizador de lagrange β. Obtemos, então, o problema contínuo $(\overline{S}_L)^b$, que é nada mais que um problema de caminho mínimo em G com custos perturbados pela restrição relaxada.

Sabemos que, para qualquer penalizador β escolhido, a solução de $(\overline{S}_L)^b$ é uma cota inferior para a solução de $(S)^b$ e, por conseguinte, para $(\overline{P})_{\bar{z}^b}$. Buscamos o penalizador ótimo através do método do subgradiente, que consiste em, a partir de um valor inicial β_0, obter uma sequência (β_i) de multiplicadores iterativamente através da projeção do multiplicador da iteração anterior na direção do subgradiente da função objetivo do problema, que determina a direção em que se encontra o valor ótimo.

6.2 Subgradiente

Seja $(\overline{S}_L)^b$ a seguir a relaxação Lagrangeana do problema $(S)^b$ pela dualização de (18c), com $\beta \geqslant 0$.

$$\begin{aligned} (\overline{S}_L)^b \qquad \min \quad & (\mathbf{c}^t + \beta \bar{\boldsymbol{\rho}}^b \mathcal{E})x - \beta \bar{\boldsymbol{\rho}}^b D\mathbf{1} && (19a) \\ s.a. \quad & \mathcal{N}x = \mathbf{b} && (19b) \\ & x \in [0,1]^m && (19c) \end{aligned}$$

Inicialmente, consideramos $\beta = \beta_0 = 0$, resolvemos o problema e verificamos se o valor da solução melhora a cota inferior $LB_{(S)^b}$ para o problema $(S)^b$. Caso isso não ocorra ou o *gap* entre as cotas $LB_{(S)^b}$ e $UB_{(S)^b}$ seja superior à margem predeterminada, determinamos a direção do subgradiente da função objetivo para a solução encontrada e obtemos o valor de β para a próxima iteração.

Estando na iteração r, o valor de β_{r+1} é calculado segundo a fórmula:

$$\beta_{r+1} \leftarrow \beta_r + tp\frac{\left(UB_{(S)^b} - LB_{(S)^b}\right)}{\|g^r\|^2} g^r,$$

onde g^r é o subgradiente dado por

$$g^r = \frac{\partial}{\partial \beta}\left((\mathbf{c}^t + \beta\bar{\boldsymbol{\rho}}^b\mathcal{E})\bar{x}^r - \beta\bar{\boldsymbol{\rho}}^b D\mathbf{1}\right) = \bar{\boldsymbol{\rho}}^b\mathcal{E}\bar{x}^r - \bar{\boldsymbol{\rho}}^b D\mathbf{1},$$

com $\bar{x}^r$ a solução ótima encontrada na iteração r e tp sendo o *tamanho do passo* dado na direção do subgradiente de forma a obter o próximo multiplicador de Lagrange.

Se após ℓ iterações consecutivas não conseguirmos melhora na cota inferior $LB_{(S)^b}$, reduzimos o tamanho do passo na esperança de convergir para o multiplicador ótimo na próxima iteração e, consequentemente, conseguirmos reduzir o *gap* entre as cotas. Inicialmente, consideramos $tp = 2$ e , após $\ell = 10$ iterações sem melhora, atualizamos tp seguindo a fórmula:

$$tp \leftarrow \frac{7}{10} tp.$$

6.3 Atualização das cotas de $(S)^b$ e critérios de parada

A cada iteração r do subgradiente, se o valor de $\bar{\boldsymbol{\rho}}^b\mathcal{E}\bar{x}^b$ para a solução $\bar{x}^r$ encontrada é inferior ao valor $\bar{\boldsymbol{\rho}}^b D\mathbf{1}$, temos que essa solução é viável para $(S)^b$ e, portanto, o valor $\mathbf{c}^t\bar{x}^r$ (o custo do caminho encontrado) é uma cota superior para esse problema, sendo candidato à atualização de $UB_{(S)^b}$ caso a melhore. Nos demais casos, o valor da função objetivo para essa solução ainda é uma cota inferior para $(S)^b$.

Como critérios de parada, assumimos os mesmos critérios descritos para o esquema de decomposição em L para (P). Consideramos o máximo de 100 iterações e um erro relativo à cota inferior de no máximo 10^{-5}.

6.4 Atualização de LB

Assim como demonstramos, soluções para $(S)^b$ fornecem cotas inferiores para (P). Como afirmamos no Capítulo 5, soluções para o problema $(\overline{P})_{\bar{z}^b}$ também fornecem cotas inferiores. Dessa forma, de posse do valor das soluções desses dois problemas, podemos considerar o maior desses valores como candidato a atualizar a melhor cota inferior para (P).

7 ESQUEMA DE *BRANCH-AND-BOUND* EM (P)

Neste capítulo, apresentamos o esquema de *B&B* desenvolvido para resolver o problema (P) diretamente. Nesse esquema, focamos na busca por caminhos viáveis através da aplicação do algoritmo de *B&B* sobre as variáveis x do modelo linear determinístico do problema, usando diferentes critérios de ramificação e de poda de nós. Não nos preocupamos com as variáveis de cenários nesse esquema, dado que a viabilidade de uma solução pode ser verificada com apenas as informações sobre o caminho obtido.

A seguir, descrevemos as estruturas fundamentais que compõem o esquema desenvolvido. Reportamos os resultados computacionais obtidos para algumas das combinações desses métodos no Capítulo 8.

7.1 Pré-processamento

De forma a reduzir a dimensão do problema a ser resolvido e, consequentemente, diminuir o tempo de resolução, aplicamos um pré-processamento no grafo de entrada de forma a identificar, em uma etapa inicial, arcos que definitivamente não pertencem a nenhuma solução ótima para (P).

Aplicamos duas técnicas de pré-processamento: uma baseada nos custos e outra baseada nos atrasos dos arcos para cada cenário. A ideia é verificar se o menor caminho (segundo esses critérios de pesos citados) entre s e t que passa por cada um dos arcos do grafo é um caminho viável de (P) através da análise de um limite inferior para o peso desse caminho. A Proposição 4 exibe esse limite, demonstrando sua corretude.

Proposição 4. *Seja f uma função de pesos sobre os arcos de um grafo $G = (N, A)$ e suponha que sua imagem é não-negativa. Defina $F(u \rightsquigarrow v) := \sum_{ij \,\in\, u \rightsquigarrow v} f(ij)$ como o peso, com relação a f, de um caminho qualquer $u \rightsquigarrow v$ em G. Sejam $s, t \in N$ tais que $s \neq t$ e $uv \in A$ tal que $uv \neq st$. Sejam $s \overset{\min}{\rightsquigarrow} u$ e $v \overset{\min}{\rightsquigarrow} t$ respectivamente os caminhos mínimos, com relação aos pesos dados por f, entre s e u e entre v e t. Tome $s \overset{uv}{\rightsquigarrow} t$ como um caminho mínimo, com relação a f, dentre os caminhos em $s \Rrightarrow t$ nos quais uv figura como um de seus arcos. Temos que $F(s \overset{\min}{\rightsquigarrow} u) + f(uv) + F(v \overset{\min}{\rightsquigarrow} t) \leqslant F(s \overset{uv}{\rightsquigarrow} t)$.*

Demonstração. Observe que, como $uv \in s \overset{uv}{\rightsquigarrow} t$, o valor de $F(s \overset{uv}{\rightsquigarrow} t) - f(uv)$ é exatamente igual à soma dos pesos dos subcaminhos de s a u e de v a t em $s \overset{uv}{\rightsquigarrow} t$. Sejam $s \overset{uv}{\rightsquigarrow} u$ e $v \overset{uv}{\rightsquigarrow} t$ esses subcaminhos. Então temos, por definição:

$$F(s \overset{\min}{\rightsquigarrow} u) \leqslant F(s \overset{uv}{\rightsquigarrow} u)$$
$$F(v \overset{\min}{\rightsquigarrow} t) \leqslant F(v \overset{uv}{\rightsquigarrow} t)$$

portanto,

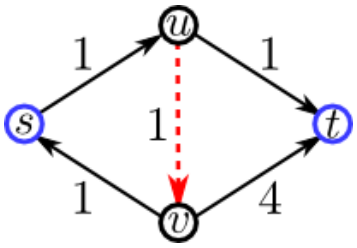
$$\begin{aligned} F(s \overset{\min}{\rightsquigarrow} u) + F(v \overset{\min}{\rightsquigarrow} t) &\leqslant F(s \overset{uv}{\rightsquigarrow} u) + F(v \overset{uv}{\rightsquigarrow} t) \\ F(s \overset{\min}{\rightsquigarrow} u) + f(uv) + F(v \overset{\min}{\rightsquigarrow} t) &\leqslant F(s \overset{uv}{\rightsquigarrow} u) + f(uv) + F(v \overset{uv}{\rightsquigarrow} t) \\ \therefore \quad F(s \overset{\min}{\rightsquigarrow} u) + f(uv) + F(v \overset{\min}{\rightsquigarrow} t) &\leqslant F(s \overset{uv}{\rightsquigarrow} t) \end{aligned}$$

□

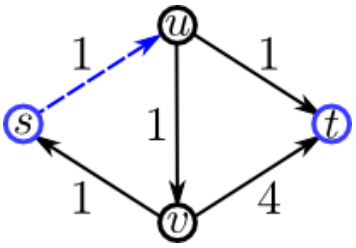
Perceba que quando os caminhos $s \overset{\min}{\rightsquigarrow} u$ e $v \overset{\min}{\rightsquigarrow} t$ não compartilham arestas, temos necessariamente que a relação demonstrada na Proposição 4 é satisfeita na igualdade. Entretanto, quando esse não é o caso, podemos ter um valor estritamente inferior, como ilustrado pela Figura 9.

Figura 9 – Exemplo de caso em que a sobreposição entre $s \overset{\min}{\rightsquigarrow} u$ e $v \overset{\min}{\rightsquigarrow} t$ leva à desigualdade estrita $F(s \overset{\min}{\rightsquigarrow} u) + f(uv) + F(v \overset{\min}{\rightsquigarrow} t) < F(s \overset{uv}{\rightsquigarrow} t)$

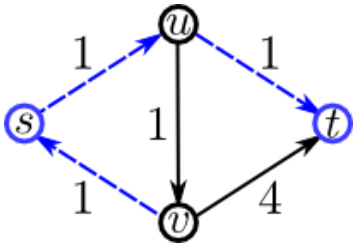
(a) Grafo e seus custos, com destaque pontilhado para o arco uv.



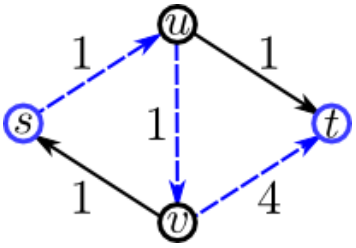
(b) Caminho $s \overset{\min}{\rightsquigarrow} u$ tracejado, com custo $f(s \overset{\min}{\rightsquigarrow} u) = 1$.



(c) Caminho $v \overset{\min}{\rightsquigarrow} t$ tracejado, com custo $f(v \overset{\min}{\rightsquigarrow} t) = 3$.



(d) Caminho $s \overset{uv}{\rightsquigarrow} t$ tracejado, com custo $f(s \overset{uv}{\rightsquigarrow} t) = 6$.



Fonte: própria autoria

Observe que $F(s \overset{\min}{\rightsquigarrow} u) + f(uv) + F(v \overset{\min}{\rightsquigarrow} t) = 1 + 1 + 3 = 5 < 6 = F(s \overset{uv}{\rightsquigarrow} t)$.

A primeira técnica também atua na busca por um limite superior para o problema, que chamaremos de UB, e este, por sua vez, tem papel na poda de nós no esquema de *B&B* e na eliminação de arcos no próprio pré-processamento.

Na primeira técnica, consideramos os caminhos mínimos com relação aos custos c dos arcos. Assim, fazendo uso da mesma notação apresentada na proposição, considerando como f a função c de custos dos arcos (com a respectiva função de peso dos caminhos C),

comparamos para cada arco $uv \in A$ o valor de $C(s \overset{\min}{\rightsquigarrow} u) + c(uv) + C(v \overset{\min}{\rightsquigarrow} t)$ com o limite UB. Caso esse valor seja superior ao limite, isso significa que o menor caminho de s a t que passa por uv não é um caminho viável para o problema. Dessa forma, marcamos uv para uma posterior remoção do grafo. Caso $C(s \overset{\min}{\rightsquigarrow} u) + c(uv) + C(v \overset{\min}{\rightsquigarrow} t) < UB$, verificamos se o conjunto $s \overset{\min}{\rightsquigarrow} u \cup \{uv\} \cup v \overset{\min}{\rightsquigarrow} t$ é um caminho e se esse caminho é viável para (P), nos permitindo atualizar o limite superior em caso afirmativo.

Na segunda técnica, consideramos, para cada cenário $k \in K$, os caminhos mínimos $s \overset{k}{\rightsquigarrow} u$ e $v \overset{k}{\rightsquigarrow} t$ com relação aos atrasos d^k dos arcos. Usando a mesma notação da Proposição 4, considerando f como a função de atrasos d^k (com a respectiva função de peso dos caminhos D^k) para cada cenário k, comparamos o valor calculado de $D^k(s \overset{k}{\rightsquigarrow} u) + c(uv) + D^k(v \overset{k}{\rightsquigarrow} t)$ com o limite de atraso D. Contabilizamos o valor $\bar{\rho}_{uv}$ como sendo a soma das probabilidades de todos os cenários em que o limite máximo de atraso é suplantado para o arco uv e, caso $\bar{\rho}_{uv} > \alpha$, sabemos que os caminhos de menor atraso de s a t que passam por uv não são viáveis. Marcamos, então, o arco uv para posterior remoção do grafo.

Após o pré-processamento, eliminamos do grafo todos os arcos marcados para remoção. A informação sobre os custos dos caminhos $s \overset{\min}{\rightsquigarrow} u$, $v \overset{\min}{\rightsquigarrow} t$, $s \overset{k}{\rightsquigarrow} u$ e $v \overset{k}{\rightsquigarrow} t$ para cada $uv \in A$ e cada $k \in K$ são armazenadas para posterior uso em um dos critérios de poda, como veremos na Seção 7.5.2. O cálculo dos caminhos mínimos é feito através do uso do algoritmo de Dijkstra e, por isso, atualizar o grafo a cada remoção de um arco tornaria o processo bastante custoso.

Observamos que a ideia por trás dessas técnicas também é descrita de forma semelhante em (DUMITRESCU; BOLAND, 2003), também com uso da função de custo e da função de atrasos, porém aplicada sobre o problema CMRL.

7.1.1 Extensão à fase de pré-processamento

Como afirmamos anteriormente, se fixarmos $z = \bar{z}$ em (P), com $\bar{z}$ satisfazendo a restrição (10d), obtemos uma instância do problema CMRL. A solução ótima para o problema $(P)_{\bar{z}}$ fornece uma cota superior para a solução de (P). Ainda, afirmamos que uma solução ótima para (P) reside em $\bar{z}$ maximais quanto à ordem $\leqslant$. Por conta disso, decidimos tentar obter cotas superiores para a solução do problema através da aplicação de uma heurística de busca de soluções viáveis para o problema $(P)_{\bar{z}}$, onde escolhemos um $\bar{z}$ maximal através do Algoritmo 1. A heurística escolhida foi a H_MCOP, introduzida por Korkmaz e Krunz (2001a).

O algoritmo H_MCOP consiste em considerar pesos não-lineares g_λ para os caminhos no grafo, definidos em função das restrições dos recursos, e tentar minimizar esses pesos através de modificações do algoritmo de Dijkstra de forma a buscar soluções para o problema.

A função g_λ é definida como

$$g_\lambda(u \rightsquigarrow v) := \sum_{\ell=1}^{|K|} \left(\frac{w_\ell(u \rightsquigarrow v)}{D_\ell} \right)^\lambda,$$

sendo $\lambda \geqslant 1$ e com

$$w_\ell(u \rightsquigarrow v) := \sum_{ij \,\in\, u \rightsquigarrow v} d_{ij}^{\ell},$$

segundo a notação do problema CMRL, introduzido na Seção 2.2.2.

Através de uma variação do algoritmo de Dijkstra, a heurística busca minimizar a função de custo g_λ para $\lambda = 1$ através da determinação dos caminhos mínimos entre u e t, $\forall\, u \in N$. Então, através de outra variação do algoritmo de Dijkstra, tenta, a partir de s, alcançar cada nó u baseando-se na minimização de g_λ, com λ dado. O caminho $s \rightsquigarrow t$ é heuristicamente construído concatenando $s \rightsquigarrow u$ com o segmento $u \rightsquigarrow t$ estimado anteriormente, observando os nós já alcançados. Se algum dos caminhos $u \rightsquigarrow t$ determinar uma composição viável, o algoritmo passa a considerar aquele que minimiza a função de custos c em vez da função não-linear g_λ, de forma a tentar obter um caminho viável de menor custo. Ao alcançar o nó t, o algoritmo verifica se o caminho obtido é viável, retornando seu custo. Caso não o seja, retorna $+\infty$ como cota.

Com o valor retornado pela heurística como cota superior, repetimos a etapa de pré-processamento até que mais nenhum arco possa ser marcado para remoção. Dessa forma, introduzimos uma certa dinamicidade sobre o grafo, onde a cada nova iteração da etapa observamos os caminhos entre os nós considerando apenas os arcos que possam pertencer à solução do problema.

Após a execução dessa extensão, atualizamos as distâncias entre os nós armazenadas, de forma a refletir o novo grafo obtido.

7.2 Critérios de ramificação

Todas as estratégias apresentadas aplicam-se ao espaço de busca das variáveis x, dado que a viabilidade de um caminho, representado pelo vetor $\bar{x}$, pode ser determinada sem a informação da solução $\bar{z}$ associada, bastando contabilizar os atrasos desse caminho para cada cenário, obtidos pelo vetor $\mathcal{E}\bar{x}$, verificando se as probabilidades dos cenários nos quais $\bar{x}$ ultrapassa o limite D somam um valor inferior à margem de risco α. Dessa forma, justificamos a escolha de escopo da ramificação nesse esquema.

Três alternativas ao critério de ramificação foram implementadas, de forma a tentar aumentar a eficiência da resolução do problema buscando a redução do número de nós ramificados até o encontro da solução ótima. Antes de iniciarmos a descrição dos critérios de ramificação, reportamos um caso de fixação trivial de variáveis aplicado ao nosso esquema de *Branch-and-Bound*.

7.2.1 Fixação trivial

Para cada variável x_{ij}, associada ao arco $ij \in A$, que seja fixada em 1 em um dado nó n^b da árvore de ***B&B***, podemos trivialmente fixar em 0 todas as variáveis associadas aos

outros arcos que *saem* de i e aos outros arcos que *chegam* em j. Dizemos que um arco $ij \in A$ *sai* de um nó $v \in N$ caso $i = v$. Em contrapartida, o arco ij *chega* em v caso $j = v$.

Tais fixações são triviais por conta da própria definição de caminho. Essas fixações desempenham um papel importante no esquema de ***B&B*** quando usamos o critério de ramificação por avanço de fronteira, descrito a seguir, de forma a garantir a obtenção de caminhos em todas as soluções inteiras encontradas. Além disso, permite a redução do número de variáveis livres no modelo, diminuindo a dimensão do problema a ser resolvido.

7.2.2 Dicotomia do mais fracionário

Como um dos critérios de ramificação para os nós da árvore de *Branch-and-Bound*, usamos o critério de *dicotomia do mais fracionário*, que consiste em, dado um nó n^b e sua solução associada $\bar{x}^b$, determinar a componente i em que o valor da solução mais se aproxima de $\frac{1}{2}$, ou seja, que mais dista de um valor inteiro. Tal componente pode ser obtida através da fórmula

$$i \in \arg\min_j \left| \frac{1}{2} - \bar{x}^b_j \right|.$$

Após a determinação da componente i, passamos a ramificar o nó n^b em dois novos nós n^b_0 e n^b_1 associados ao mesmo problema local de n^b com a adição da fixação em 0 e em 1, respectivamente, da variável x_i. Dessa forma, no ramo enraizado em n^b_0 buscamos soluções onde o arco representado por x_i não faz parte dos caminhos obtidos e no ramo enraizado em n^b_1 buscamos soluções onde esse arco obrigatoriamente faz parte.

Perceba que uma solução inteira $\bar{x}^b$, obtida em um nó n^b ao utilizarmos essa estratégia de ramificação, pode não ser viável para o problema (P). Assim, se no caminho $s \rightsquigarrow t$ representado por $\bar{x}^b$ ainda constarem arcos cujas variáveis associadas não foram fixas em n^b, devemos continuar a busca por soluções do problema ramificando essas variáveis, desde que o custo de $\bar{x}^b$ não seja superior ao da melhor solução viável já encontrada.

Como em $\bar{x}^b$ só há valores inteiros, não faz sentido aplicarmos a dicotomia do mais fracionário. Nessas situações, o que fazemos é ramificar todas as variáveis não-fixas de valor unitário seguindo a estratégia de (FREITAS, 2011).

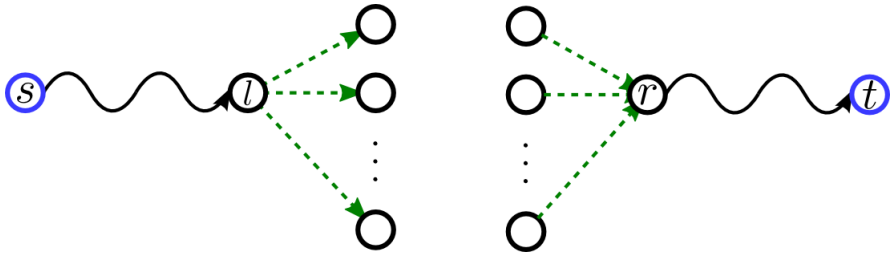
7.2.3 Avanço de fronteira

A técnica de *avanço de fronteira* consiste na mesma ideia da técnica anterior, entretanto com um número restrito de candidatos a ramificação. Ao contrário da técnica de dicotomia do mais fracionário, não verificamos os valores de todas as componentes no vetor-solução $\bar{x}^b$, mas apenas aqueles que nos permitem o "avanço de fronteira". A cada ramificação realizada, estendemos subcaminhos obtidos a partir de s ou com destino em t através da adição de um novo arco a uma de suas *extremidades livres*, ou, no ramo de contrapartida, desconsideramos a presença desse arco nas soluções.

Se observarmos o grafo $\overline{G}$ induzido pelos arcos associados a componentes de valor unitário do vetor $\bar{x}^b$, associado à solução no nó n^b da árvore de *B&B*, chamamos de extremidades livres o nó l mais distante de s e o nó r mais distante de t, em número de arcos, tal que existem caminhos $s \rightsquigarrow l$ e $r \rightsquigarrow t$. Ao nó l chamamos de *extremidade esquerda* e ao nó r chamamos de *extremidade direita*.

Dessa forma, podemos perceber que as únicas variáveis que nos permitem estender os subcaminhos $s \rightsquigarrow l$ e $r \rightsquigarrow t$ são aquelas associadas aos arcos que saem de l e aos arcos que chegam em r. Buscamos, então, a componente mais fracionária apenas dentre as associadas a essas variáveis, a fim de ramificarmos o nó n^b. A Figura 10 ilustra o critério de ramificação por avanço de fronteira, onde as setas onduladas entre s e l e entre r e t abstraem arcos cujas variáveis assumiram valor unitário na solução do nó n^b.

Figura 10 – Visualização da etapa de ramificação.

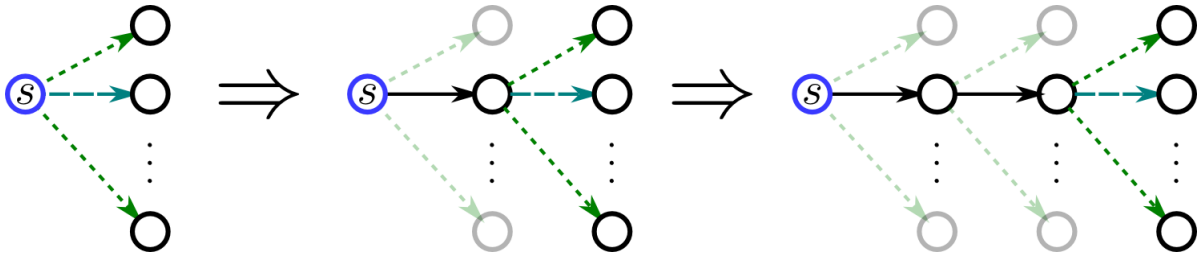


Fonte: própria autoria

Os arcos destacados tracejados são considerados na escolha da variável a ramificar.

A Figura 11 ilustra abstratamente como o caminho solução é progressivamente construído através do avanço da fronteira dos subcaminhos a cada nó do *Branch-and-Bound*.

Figura 11 – Exemplo de avanço de fronteira a partir de s, etapa por etapa.



Fonte: própria autoria

A cada etapa, o arco escolhido está destacado diferentemente dos demais.

Assim, dado um nó n^b e sua solução associada $\bar{x}^b$, determinamos as extremidades l e r e escolhemos a componente i a ramificar dentre os arcos incidentes a esses nós. Entretanto, perceba que é possível que nos subcaminhos $s \rightsquigarrow l$ e $r \rightsquigarrow t$ ocorram arcos cujas variáveis associadas não estão fixas no ramo da árvore de *B&B* enraizado por n^b. Nesse caso a ramificação

do nó n^b sem a adição de fixação dessas variáveis pode culminar, em iterações posteriores, em soluções de outros nós desse ramo com valores fracionários para essas variáveis.

Para eliminar essa possibilidade, ramificamos esse nó considerando também essas variáveis, procedendo da seguinte forma: determinamos o conjunto F de variáveis não-fixas em n^b cujos arcos correspondentes pertencem a algum dos subcaminhos $s \rightsquigarrow l$ e $r \rightsquigarrow t$ e aplicamos a estratégia de ramificação de Freitas (2011) sobre esse conjunto de variáveis. Para essa ramificação, consideramos a ordem $\preceq$ em que todas as variáveis dos arcos do subcaminho $s \rightsquigarrow l$ precedem as do subcaminho $r \rightsquigarrow t$, e onde as variáveis dos arcos em um mesmo subcaminho são ordenadas do mais próximo ao mais distante, em número de arcos, ao nó s ou t (dependendo do subcaminho em questão).

Por conta dessa ordem e da forma como a estratégia de (FREITAS, 2011) escolhe as fixações de variáveis, a cada nó ramificado consideramos o caso em que um arco uv de $s \rightsquigarrow l$ ou de $r \rightsquigarrow t$ não faz parte da solução e de forma que, se $uv \in s \rightsquigarrow l$, os subcaminhos $s \rightsquigarrow u$ e $r \rightsquigarrow t$ contêm apenas arcos com as respectivas variáveis fixas em 1 e, caso $uv \in r \rightsquigarrow t$, essa situação ocorre para os subcaminhos $s \rightsquigarrow l$ e $v \rightsquigarrow t$. Por fim, aplicamos o avanço de fronteira, fixando todas as variáveis de F em 1 nos respectivos nós.

7.2.4 Limite superior generalizado

O critério de ramificação de *limite superior generalizado (LSG)* é uma versão do critério de avanço de fronteira onde ramificamos os nós por dicotomia em um subconjunto de variáveis em vez de em uma única.

Esse critério consiste em, dado um nó n^b da árvore de *B&B*, selecionar um conjunto de índices I de variáveis x não-fixas em n^b forma que a igualdade

$$\sum_{i \in I} x_i = 1$$

seja válida para as variáveis em questão (ou seja, exatamente uma dentre x_i com $i \in I$ deve ser não-nula). Dado o vetor-solução fracionário $\bar{x}^b$ desse nó, particionamos I em dois subconjuntos I_1 e I_2 de tal maneira que $\sum_{i \in I_1} \bar{x}_i^b$ e $\sum_{j \in I_2} \bar{x}_j^b$ sejam aproximadamente de mesmo valor. Então, dois novos nós são ramificados, um com a adição do corte

$$\sum_{i \in I_1} x_i = 0$$

e o outro com o corte

$$\sum_{j \in I_2} x_j = 0.$$

Neste esquema de *B&B*, decidimos utilizar o avanço de fronteira partindo apenas do nó origem s, de forma que o conjunto de índices usado na aplicação dos cortes LSG é o das variáveis associadas aos arcos que saem da extremidade esquerda para cada nó n^b em questão. Esse conjunto é válido para a aplicação do critério, já que exatamente um desses arcos deve

ser escolhido para formar o caminho desejado. Também procedemos como no critério anterior quanto à eliminação de arcos do subcaminho $s \rightsquigarrow l$ cujas variáveis não estejam fixas.

7.3 Seleção de nós

Assim como no esquema de *Branch-and-Bound* desenvolvido para a resolução do problema (M), neste esquema mantemos uma lista de nós ativos, armazenados em uma lista de prioridades que denotamos por $\mathcal{H}$. Essa lista está organizada de forma que, dados dois nós ativos n^a e n^b, com os respectivos valores ${}^t\bar{x}^a$ e ${}^t\bar{x}^b$ das soluções de seus problemas associados, temos que n^a tem maior prioridade sobre n^b caso ${}^t\bar{x}^a < {}^t\bar{x}^b$. No caso em que esses valores são iguais, determinamos que o nó com o maior número de variáveis fixas tem maior prioridade sobre o outro. O uso desse critério de desempate se mostrou notável na redução do número de nós explorados em alguns dos testes realizados.

A cada iteração do algoritmo de *B&B* removemos de $\mathcal{H}$ o nó de maior prioridade, ramificando-o e adicionando os novos nós obtidos à lista de prioridades.

7.4 Atualização de limites

O menor dentre os valores das soluções encontradas em nós n^b ativos fornece uma cota inferior para a solução de (P), dado que a cada n^b resolvemos problemas que diferem de (P) apenas por fixações de variáveis. Dessa forma, a cada iteração do algoritmo de *B&B* temos que o valor da solução do nó de maior prioridade em $\mathcal{H}$ é uma cota inferior para (P) e, caso esse valor seja superior à melhor cota já obtida até o momento, que representamos por LB, atualizamos essa cota.

Para nós cujo vetor-solução $\bar{x}^b$ seja inteiro, verificamos se o vetor de cenários $\bar{z}$, calculado através do método descrito no início deste capítulo, é tal que a soma das probabilidades dos cenários violados é inferior à margem de risco α. Em caso afirmativo, o valor da solução $\bar{x}^b$ obtida fornece cota superior para o problema, já que $\bar{x}^b$ representa um caminho viável de (P). Em caso negativo, esse valor ainda pode configurar uma cota inferior, desde que seja o menor dentre todos os nós ativos. Quando confirmada a validade da cota superior, atualizamos a melhor cota UB já obtida até o momento, caso seja possível.

7.5 Critérios de poda

7.5.1 Poda por limites

Dado um nó n^b, caso o valor de sua solução $\bar{x}^b$ seja tal que $\mathbf{c}^t\bar{x}^b > UB$, podemos remover da árvore de *B&B* todo o ramo enraizado por esse nó, pois, como $\mathbf{c}^t\bar{x}^b$ é uma cota inferior local para as soluções desse ramo, concluímos que nenhuma das possíveis soluções viáveis fornecidas por esses nós seria melhor que as soluções já encontradas.

Outra situação que permite a poda da árvore de *B&B* é quando encontramos uma solução viável de (P) cujo valor melhora a cota superior UB. Podemos eliminar todos os nós da lista de nós ativos cujo valor da solução supere o novo valor de UB, pelo mesmo motivo exposto anteriormente.

7.5.2 Poda por violação antevista

Através do uso das informações armazenadas na fase de pré-processamento, somos capazes de reconhecer antecipadamente ramos da árvore de *B&B* que certamente não nos levarão à solução ótima de (P).

Supomos que armazenamos as informações de custos mínimos nos vetores $\mathcal{R}_s$, $\mathcal{R}_t$, onde as entradas $\mathcal{R}_s(u)$ e $\mathcal{R}_t(v)$ guardam respectivamente os valores do menor caminho entre s e u e entre v e t em G, e as informações de atrasos mínimos nos vetores $\mathcal{T}_s^k$ e $\mathcal{T}_t^k$, $\forall k \in K$, onde as entradas $\mathcal{T}_s^k(u)$ e $\mathcal{T}_t^k(v)$ guardam respectivamente os menores atrasos possíveis entre s e u e entre v e t no cenário k. Suponha que nas entradas referentes a nós para os quais não haja caminho em G conste o valor ∞. Assim, dado um nó n^b da árvore e seus subcaminhos $s \rightsquigarrow l$ e $r \rightsquigarrow t$ obtidos por avanço de fronteira, podemos antever se a solução ótima inteira do ramo enraizado em n^b é inviável para (P) quando ocorrerem os seguintes casos:

- $c(s \rightsquigarrow l) + \mathcal{R}(l,t) > UB$, ou
- $\mathcal{R}(s,r) + c(r \rightsquigarrow t) > UB$, ou
- a soma das probabilidades dos cenários k tais que $d^k(s \rightsquigarrow l) + \mathcal{T}^k(l,t) > D$ supera α, ou
- a soma das probabilidades dos cenários k tais que $\mathcal{T}^k(s,r) + d^k(r \rightsquigarrow t) > D$ supera α.

Quando alguma dessas situações é confirmada, podemos seguramente concluir que a solução ótima para (P) não reside nas soluções dos nós nesse ramo, permitindo-nos podá-lo, dado que as primeiras situações denunciam soluções que não melhoram a cota superior UB e as demais denunciam soluções inviáveis para o problema. No caso em que usamos avanço de fronteira apenas a partir do nó origem s, consideramos apenas as situações que analisam o subcaminho $s \rightsquigarrow l$. Como poderemos ver no Capítulo 8, a adoção desse critério de poda no algoritmo de *B&B* mostrou-se bastante eficaz de uma maneira geral, reduzindo o tempo necessário à resolução de algumas instâncias em até três vezes.

7.6 Critério de parada

Como critérios de parada, consideramos os seguintes casos:

- conseguimos provar a otimalidade de uma solução viável através da igualdade $LB = UB$,
- a lista de nós ativos tornou-se vazia e todo o espaço de soluções viáveis foi inspecionado,
- o erro absoluto entre as melhores cotas encontradas é inferior a uma margem informada,
- o erro relativo entre essas cotas é inferior a uma margem informada.

Caso a lista de nós ativos tenha esvaziado e não tenhamos encontrado nenhuma solução viável para (P), isso significa que esse problema é inviável. Em todos os outros casos, a solução ótima reside na melhor solução viável encontrada.

8 RESULTADOS COMPUTACIONAIS

Reportamos neste capítulo os resultados dos testes computacionais realizados com as técnicas desenvolvidas e descritas em capítulos anteriores.

Na próxima seção discutimos aspectos de implementação e execução importantes presentes nos programas usados para gerar os resultados. Nas seções subsequentes mostramos tabelas comparativas entre os diferentes métodos de resolução desenvolvidos. Na última seção exibimos os resultados da melhor estratégia obtida em comparação com os resultados da ferramenta de resolução de problemas de otimização linear IBM ILOG CPLEX®[1]. A cada seção, apenas as informações mais relevantes ao comparativo são inclusas nas tabelas. Ao longo do texto, chamamos atenção para certos aspectos importantes e interpretações dos resultados que podem ser inferidas dessas informações.

Cada resultado reportado configura as informações coletadas em uma execução da referida estratégia. Parte dos resultados exibidos aqui foram apresentados no XIII ROADEF em 2012 (ANDRADE; ARARUNA; LISSER, 2012).

8.1 Questões de implementação

Fazemos uso da linguagem de programação C++ na implementação de todas as estratégias desenvolvidas, gerando o código-objeto correspondente através do compilador `g++`, do pacote GNU COMPILER COLLECTION[2]. Os testes foram executados em um computador dotado de dois processadores INTEL XEON® 3.0GHz e 4GB de memória sob a arquitetura de 32bits. Toda a otimização de modelos lineares foi realizada com o auxílio da ferramenta CPLEX na versão 12.5 e sua interface de programação CONCERT TECHNOLOGY C++ API. Não fazemos uso do recurso de paralelismo nas resoluções dos modelos, oferecido pelo CPLEX, de forma a obtermos comparativos justos e determinísticos.

8.1.1 Nós rejeitados

Em nossos experimentos iniciais, usamos um *heap* como estrutura de dados para representar a lista de prioridades onde armazenamos os nós ativos do esquema de *B&B*. Entretanto, trata-se de uma estrutura de tamanho estático, o que significa que, uma vez instanciada, não podemos ampliar ou reduzir seu tamanho sem necessitarmos da movimentação de todos os dados armazenados para uma nova área da memória. Esse tipo de operação é bastante custosa.

Como não temos meios de determinar o número máximo de nós que estarão presentes na lista de ativos simultaneamente em uma execução do algoritmo, não temos como determinar o melhor tamanho a ser usado para o *heap*. Dessa forma, podem ocorrer casos em que

[1] `http://www.ibm.com/software/commerce/optimization/cplex-optimizer/`
[2] `http://gcc.gnu.org/`

a lista de nós ativos está com sua capacidade máxima de elementos mas necessitamos adicionar novos nós em decorrência de uma ramificação. Para evitar a realocação da estrutura, operação que poderia ter que ser realizada diversas vezes no decorrer da execução do programa, consideramos os elementos excedentes como *nós rejeitados*. Isso significa que os ramos enraizados por esses nós na árvore de *B&B* não serão explorados e os referidos nós serão descartados.

Como é de se esperar, ao tomarmos tal decisão corremos o risco de descartar exatamente o ramo que nos levaria à solução ótima do problema. Por conta disso, mantemos a informação sobre o menor dos valores das soluções de nós rejeitados, que configuram cotas inferiores locais, de maneira que a cota inferior global para (P) é atualizada seguindo $LB \leftarrow \min\{LB_{\mathcal{H}};\ LB_r\}$ a cada iteração, onde $LB_{\mathcal{H}}$ é o valor da solução do nó de maior prioridade em $\mathcal{H}$ e LB_r é a menor cota dentre os nós rejeitados. Com isso, somos capazes de reconhecer os casos em que a solução ótima possa ter sido descartada, dados pelo momento em que $LB = LB_r$. Assim, devemos notar que, para esses experimentos, caso algum nó tenha sido rejeitado não podemos afirmar que a melhor solução encontrada é de fato a solução ótima mesmo tendo esvaziado a lista de nós ativos.

De forma a eliminar essa deficiência, para os últimos experimentos realizados substituímos essa estrutura de dados por uma fila de prioridades dinâmica, onde nenhuma capacidade máxima teórica é esperada, organizada como um *heap* binomial. Dessa forma, em se tendo disponibilidade de tempo e memória suficientes, seremos capazes de obter as soluções ótimas de todas as instâncias sempre que essas existirem.

8.1.2 Número de cenários

Como argumentamos no Capítulo 3, considerar K como o conjunto de todos os cenários de atrasos possíveis para os arcos de G culmina na equivalência entre o modelo determinístico (P) e a formulação estocástica (7). Porém, considerar todos esses cenários é inviável na prática, ao percebermos o número de restrições com as quais teríamos que lidar no modelo a ser otimizado. Assim, usamos o artifício de considerar apenas um subconjunto de K com cardinalidade significativa. Optamos por verificar apenas conjuntos com 50 elementos, selecionados ao acaso, para todos os testes reportados neste capítulo. A seleção ao acaso de cada cenário é abstraída em sua constituição, feita através de uma amostra de cada variável aleatória associada aos arcos de G. Dessa forma, tentamos conciliar a viabilidade computacional e a qualidade das soluções obtidas.

A forma como calculamos as probabilidades dos cenários considerados está descrita no Apêndice B.

8.1.3 Margem de risco

Em todos os testes optamos por considerar a margem de risco $\alpha = 5\%$, dado que essa margem é a mais utilizada por projetistas em seus critérios de *QoS*, por fornecer uma boa contrapartida entre economia de custos e ônus por não-conformidade.

8.1.4 Atraso máximo

Para todas as instâncias de teste reportadas, determinamos o menor valor de limite de atraso D de forma que o problema ainda possua solução. Assim, exibimos os resultados de tempo para essas instâncias nos baseando nesse valor. É razoável crer que, para esses valores, temos bons casos de teste para os métodos desenvolvidos, por estarmos verificando valores minimais de solução.

8.2 Soluções

Na Tabela 1, sob a coluna "CM", exibimos os custos dos caminhos mínimos clássicos (sem restrições de atraso) para nossas instâncias de teste. A descrição completa das instâncias usadas pode ser encontrada no Apêndice A. Os custos dos caminhos ótimos para o problema CMRPAM constam sob a coluna "CMRPAM". Os valores nessa coluna foram obtidos através da submissão da formulação determinística do problema para cada instância ao programa CPLEX, coletando o valor reportado como custo da solução ótima encontrada. Para a instância 45 o programa não foi capaz de encontrar uma solução ótima dentro do período de 24h imposto como tempo limite de execução. A coluna "D" exibe o limite de atraso considerado para as soluções do problema.

8.3 Enumeração de partições

O método de resolução da decomposição de (P) através de *Branch-and-Bound* não se mostrou eficiente, ultrapassando o tempo limite de 24h de execução para todas as instâncias, inclusive as de menor dimensão. Podemos justificar essa demora por conta do fato de, a cada nó n^b selecionado, solucionamos o problema inteiro $(M)^b$ para só então, de posse da solução $\bar{z}^b$, podermos buscar uma solução inteira para $(P)_{\bar{z}^b}$. Dessa forma, estamos tentando otimizar mais de um problema inteiro simultaneamente, o que demanda um tempo significativo.

Por conta disso, exibimos na Tabela 2 as melhores cotas encontradas dentro do tempo limite estabelecido apenas para as menores instâncias de testes, de forma a ilustrar a dificuldade do método em resolver o problema. As colunas "UB" e "LB" exibem respectivamente as cotas inferiores e superiores e a coluna "Tempo CPU UB" informa quanto tempo levamos até encontrarmos a melhor cota superior reportada.

Perceba que para todas as instâncias o método foi capaz de encontrar uma solução de custo ótimo, entretanto não conseguimos provar sua otimalidade dentro do tempo limite devido ao *gap* entre as cotas. Ainda assim, o tempo necessário para determinar essas soluções foi bastante alto, chegando próximo ao tempo limite para a instância 10.1.

Tabela 1 – Custos dos caminhos mínimos e das soluções ótimas para o problema CMRPAM para as instâncias de teste.

Instância	D	CM	CMRPAM
10.1	38	187	350
10.2	38	91	101
10.3	52	131	188
10.4	39	100	189
10.5	48	111	171
12.1	39	305	441
12.2	57	116	192
12.3	52	124	182
12.4	58	83	231
12.5	63	165	242
16.1	51	302	662
16.2	108	268	373
16.3	99	230	394
16.4	67	245	320
16.5	90	252	311
21.1	207	211	298
21.2	78	203	395
21.3	94	293	541
21.4	109	272	453
21.5	103	243	331
24	115	424	896
29	138	549	916
33	165	713	1101
37	176	720	1328
45	613	592	?

Fonte: própria autoria

Tabela 2 – Resultados de tempo do *B&B* em (M) para as menores instâncias de teste.

Instância	UB	LB	Tempo CPU UB
10.1	350	318,589	20h35min5s345ms
12.1	441	405,86	1min3s070ms
16.1	662	499,562	2h0min45s872ms
21.1	298	222,571	2s782ms

Fonte: própria autoria

8.4 Melhora das cotas inferiores

Apesar de aplicarmos a técnica descrita no Capítulo 6 de forma a tentarmos obter melhores cotas superiores para os nós da árvore de *B&B*, continuamos não conseguindo obter as soluções ótimas para nenhuma das instâncias dentro do tempo limite de 24h. Como podemos ver na Tabela 3, a técnica utilizada não serviu a contento seu propósito de melhoria das cotas, porém conseguimos reorganizar as prioridades dos ramos que levam às soluções ótimas na lista de prioridades de nós ativos do *B&B*, de forma que exploramos esses mais antecipadamente, como podemos perceber por conta da notável diminuição do tempo necessário para encontramos as melhores cotas superiores, reportados sob a coluna "Tempo CPU UB".

Tabela 3 – Resultados de tempo do *B&B* em (M) para as menores instâncias de teste, acrescido da técnica de melhora de cotas inferiores.

Instância	UB	LB	Tempo CPU UB
10.1	350	318,762	0s760ms
12.1	441	406,093	0s872ms
16.1	662	500,581	1s683ms
21.1	298	222,577	0s293ms

Fonte: própria autoria

Ainda assim, temos dificuldade na prova de otimalidade das soluções encontradas.

8.5 *Branch-and-Bound* sobre (P)

Nessa seção apresentamos os resultados com relação ao esquema de *Branch-and-Bound* desenvolvido de forma a resolver (P) diretamente. A cada subseção, comparamos as alternativas descritas no Capítulo 7 para cada sub-rotina do esquema, determinando a mais eficiente através da análise dos parâmetros considerados nos resultados reportados. A cada subseção consideramos as estratégias mais eficientes das subseções anteriores e variamos apenas as subrotinas a comparar, de forma a podermos comparar o impacto que essas variações têm no esquema.

8.5.1 Pré-processamento

A Tabela 4 exibe a quantidade de tempo empregado na etapa de pré-processamento descrita para cada instância de teste considerada. Ambas as técnicas descritas na Seção 7.1 são consideradas simultaneamente. As colunas sob "Versão Estática" se referem aos resultados quanto ao pré-processamento sem a fase estendida, descrita na Seção 7.1.1, e as colunas sob "Versão Dinâmica" dizem respeito aos resultados quando consideramos a dinamicidade do grafo ao empregarmos a extensão da fase de pré-processamento. A coluna "Tempo CPU" exibe os tempos empregados nessas etapas. Na coluna "$|A|$" informamos a quantidade de arcos do grafo para cada instância e, sob a coluna "Arcos", informamos a quantidade de arcos removidos.

Tabela 4 – Tempos da etapa de pré-processamento, o número de arcos no grafo, o número de arcos removidos e o número de cenários pré-fixados.

Instância	$\|A\|$	Versão Estática		Versão Dinâmica		Cenários
		Arcos	Tempo CPU	Arcos	Tempo CPU	
10.1	360	181	005ms	339	013ms	4
10.2	360	337	002ms	350	001ms	5
10.3	360	300	005ms	350	007ms	2
10.4	360	193	005ms	300	014ms	1
10.5	360	261	005ms	350	008ms	4
12.1	528	387	005ms	401	006ms	2
12.2	528	366	008ms	389	017ms	3
12.3	528	364	008ms	373	015ms	4
12.4	528	386	009ms	450	017ms	3
12.5	528	412	009ms	419	012ms	2
16.1	960	668	018ms	668	023ms	4
16.2	960	660	020ms	700	040ms	3
16.3	960	644	020ms	649	030ms	4
16.4	960	391	023ms	691	044ms	2
16.5	960	479	022ms	489	092ms	3
21.1	1680	1383	041ms	1387	099ms	4
21.2	1680	1162	040ms	1183	149ms	2
21.3	1680	1258	042ms	1261	123ms	2
21.4	1680	1196	045ms	1237	150ms	4
21.5	1680	1136	044ms	1152	219ms	3
24	2208	1491	060ms	1494	085ms	3
29	3248	2232	119ms	2232	119ms	2
33	4224	3200	143ms	3200	154ms	1
37	5328	4007	188ms	4007	219ms	3
45	7920	6725	338ms	6741	380ms	1

Fonte: própria autoria

A coluna "Cenários" informa o número de cenários cuja probabilidade ultrapassa a margem de erro α e para os quais, por conta disso, devemos obrigatoriamente satisfazer a restrição de atraso máximo, ou seja, podemos fixar sua variável correspondente no valor nulo.

Não reportamos o valor da cota superior obtida através do uso da heurística H_MCOP, pois esta foi apenas capaz de determinar uma solução viável para a instância 10.2, para a qual retornou a cota ótima. Para as demais instâncias, entretanto, nenhuma solução viável foi encontrada. Acreditamos que isso se deva ao fato de termos selecionado como limite máximo de tempo o valor minimal que não torna as instâncias inviáveis, assim como descrito na Seção 8.1.4.

Perceba que para todas as instâncias conseguimos reduzir significativamente o número de arcos a considerar, tendo um impacto direto no número de variáveis do programa linear correspondente e, por conseguinte, no tempo necessário para encontrar as soluções ótimas das instâncias. Entre as versões estática e dinâmica, conseguimos diferenças modestas quanto ao número de arcos removidos do grafo, sendo a mais expressiva a obtida na instância 16.4. Para as instâncias 16.1, 29, 33 e 37 não obtivemos qualquer diferença.

Apesar disso, ao repetirmos a fase de pré-processamento na versão dinâmica, atua-

lizamos as informações de caminhos mínimos entre nós do grafo, que são utilizadas no critério de poda por violação antevista, sendo considerados apenas os arcos não removidos. Essa atualização tem impacto diretamente no número de iterações do algoritmo de *Branch-and-Bound*, como podemos perceber mais adiante.

8.5.2 Ramificação

Nessa subseção, reportamos os resultados considerando o grafo reduzido obtido após a fase de pré-processamento. Consideramos como critério de poda apenas o critério de *poda por limites*, descrito na Seção 7.5.1. As Tabelas 5 e 6 reportam os resultados analisando as três estratégias de ramificação descritas. As colunas sob "Ramificação - +Fracionário" dizem respeito à ramificação por dicotomia do mais fracionário; as colunas sob "Ramificação - +Fracionário (extremidades)" informam resultados sobre a ramificação por avanço de fronteira; por fim, as colunas sob "Ramificação - LSG" exibem os resultados da estratégia de dicotomia por Limite superior generalizado.

Na Tabela 5 exibimos as colunas "Iterações", que informam o número de iterações para cada variação, e as colunas "Nós explorados", que representam a quantidade total de nós obtidos por ramificações durante as iterações do algoritmo de *B&B*. As entradas para as estratégias que excederam o tempo limite de 24h para encontrar a solução ótima foram marcadas por um traço (–).

Observe que, apesar de mais localizada, a estratégia de dicotomia nas extremidades gera, na maioria dos casos, uma quantidade maior de nós explorados que as outras estratégias. À exceção das instâncias 16.3, 21.4 e 21.5, a estratégia de ramificação que usa limites superiores generalizados foi a que obteve um menor número de nós explorados, também conseguindo, para a instância 29, alcançar um critério de parada em um tempo inferior ao tempo limite estabelecido.

Na Tabela 6 mostramos dados sobre as soluções encontradas pelas variações analisadas nesta subseção. As colunas "Tempo CPU" dizem respeito ao tempo que cada teste desprendeu até finalizar sua execução. Entradas marcadas com TLE indicam que mesmo após 24h de execução essas estratégias não levaram à satisfação de nenhum critério de parada. As colunas "UB" e "LB" informam respectivamente os valores das melhores cotas superiores e inferiores encontradas durante a execução de cada teste. As colunas "*Status*" determinam a condição de parada satisfeita de forma a finalizar a execução do algoritmo de *B&B*. *OPT* significa que os limites superior e inferior encontrados foram idênticos; *ITER* indica que a lista de nós ativos foi esvaziada, consequentemente sinalizando que a melhor solução viável encontrada era ótima; *REJ* indica que a solução ótima foi eventualmente perdida em nós rejeitados pela lista de nós ativos durante a execução do algoritmo.

Podemos ver que, em geral, a estratégia de dicotomia do mais fracionário nas extremidades se mostra menos eficiente que a dicotomia em qualquer variável. Isso se deve pela diferença no número de nós explorados por cada uma dessas variações e no número de iterações

Tabela 5 – Comparativo entre as estratégias de ramificação do esquema de *B&B* em (P) quanto à manipulação de nós do *B&B*.

Instância	Ramificação - +Fracionário		Ramificação - +Fracionário (extremidades)		Ramificação - LSG	
	Iterações	Nós explorados	Iterações	Nós explorados	Iterações	Nós explorados
10.1	62	205	55	262	57	199
10.2	2	21	2	41	1	21
10.3	17	104	17	178	17	104
10.4	305	1473	257	2449	282	1428
10.5	35	106	22	129	25	97
12.1	81	374	64	505	80	324
12.2	241	842	187	1120	206	751
12.3	218	890	210	1020	218	755
12.4	105	509	90	878	93	507
12.5	126	546	103	769	111	503
16.1	3149	15432	2669	11736	2659	11293
16.2	830	4288	722	4915	813	4244
16.3	1049	3505	569	3756	1079	4325
16.4	603	4371	725	5831	818	4123
16.5	774	4183	619	5875	691	3354
21.1	2136	10752	2045	9709	2129	9527
21.2	296667	1653826	272542	2236138	234530	1254310
21.3	364372	2029604	240850	1958995	301060	1412440
21.4	9682	40959	4439	32122	10469	49629
21.5	204	694	146	874	222	897
24	729274	3980042	–	–	420178	2028133
29	–	–	400579	3427300	413338	2197653
33	–	–	–	–	–	–
37	–	–	–	–	–	–
45	–	–	–	–	–	–

Fonte: própria autoria

Tabela 6 – Comparativo entre as estratégias de ramificação do esquema de *B&B* em (P) quanto às soluções obtidas.

Instância	D	Ramificação - +Fracionário				Ramificação - +Fracionário (extremidades)				Ramificação - LSG			
		Tempo CPU	UB	LB	*Status*	Tempo CPU	UB	LB	*Status*	Tempo CPU	UB	LB	*Status*
10.1	38	0s365ms	350	350	OPT	1s288ms	350	350	OPT	0s258ms	350	350	OPT
10.2	38	0s014ms	101	101	ITER	0s031ms	101	101	ITER	0s007ms	101	101	ITER
10.3	52	0s051ms	188	188	ITER	0s126ms	188	188	ITER	0s045ms	188	188	ITER
10.4	39	1s474ms	189	189	OPT	2s116ms	189	189	OPT	1s106ms	189	189	OPT
10.5	48	0s095ms	171	171	ITER	0s113ms	171	171	ITER	0s703ms	171	171	ITER
12.1	39	0s785ms	441	441	ITER	1s656ms	441	441	ITER	0s312ms	441	441	ITER
12.2	57	2s089ms	192	192	ITER	1s138ms	192	192	ITER	0s844ms	192	192	ITER
12.3	52	2s371ms	182	182	OPT	4s773ms	182	182	OPT	0s889ms	182	182	OPT
12.4	58	0s920ms	231	231	ITER	2s438ms	231	231	ITER	0s428ms	231	231	ITER
12.5	63	1s066ms	242	242	OPT	2s029ms	242	242	OPT	0s424ms	242	242	OPT
16.1	51	1min38s318ms	662	662	OPT	28s840ms	662	662	OPT	23s486ms	662	662	OPT
16.2	108	40s340ms	373	373	ITER	1min1s603ms	373	373	OPT	8s938ms	373	373	ITER
16.3	99	40s989ms	394	394	OPT	1min0s678ms	394	394	OPT	10s636ms	394	394	OPT
16.4	67	37s694ms	320	320	OPT	1min29s588ms	320	320	OPT	14s063ms	320	320	OPT
16.5	90	55s193ms	311	311	OPT	2min11s723ms	311	311	OPT	10s865ms	311	311	OPT
21.1	207	1min46s871ms	298	298	OPT	36s436ms	298	298	OPT	27s761ms	298	298	OPT
21.2	78	2h8min47s604ms	398	330,577	REJ	3h5min55s822ms	395	351	REJ	1h2min34s576ms	398	317,5	REJ
21.3	94	1h54min52s200ms	541	478	REJ	2h5min30s495ms	541	502,333	REJ	52min22s175ms	541	499,571	REJ
21.4	109	4min50s095ms	453	453	OPT	4min7s351ms	453	453	OPT	3min5s542ms	453	453	OPT
21.5	103	28s554ms	331	331	ITER	32s476ms	331	331	ITER	4s675ms	331	331	ITER
24	115	15h8min0s230ms	991	696,333	REJ	TLE	–	–	–	3h21min14s734ms	896	704,5	REJ
29	138	TLE	–	–	–	TLE	–	–	–	7h25min41s138ms	978	575,667	REJ
33	165	TLE	–	–	–	TLE	–	–	–	TLE	–	–	–
37	176	TLE	–	–	–	TLE	–	–	–	TLE	–	–	–
45	613	TLE	–	–	–	TLE	–	–	–	TLE	–	–	–

Fonte: própria autoria

necessárias para finalizar a execução. Mais uma vez, confirmamos que a melhor estratégia de ramificação desenvolvida é a por dicotomia usando LSG, pois em todas as instâncias o tempo necessário foi inferior às demais, às vezes obtendo significativas reduções, como nas instâncias 21.2 e 21.3, e em outros casos conseguindo reportar resultados abaixo do tempo limite onde as outras estratégias não foram capazes, como nas instâncias 24 e 29.

8.5.3 Poda

Comparamos nas Tabelas 7 e 8 os resultados obtidos pelo esquema de *B&B* tanto utilizando apenas a poda por limites como utilizando ambos os critérios de poda descritos na Seção 7.5. Consideramos o esquema de ramificação por LSG em todos os testes nessa subseção, que se mostrou o mais eficiente como podemos ver na Seção 8.5.2.

Exibimos também os resultados ao executarmos o esquema utilizando a poda por violação antevista ao adicionarmos a etapa estendida da fase de pré-processamento, de forma a considerarmos apenas os arcos não removidos no cálculo dos caminhos mínimos entre os nós do grafo. Além disso, para essas execuções, realizamos a substituição da estrutura de dados representante da lista de nós ativos da árvore de *B&B* de forma a não nos preocuparmos mais com a rejeição de novos nós ramificados.

As colunas sob "Poda por Limites" dizem respeito aos mesmos resultados obtidos na seção anterior para a estratégia de ramificação com LSG. As colunas sob "Poda por Limites e por Violação Antecipada" apresentam os resultados para os testes onde podamos nós também pelo critério de violação antecipada descrito na Seção 7.5.2. As colunas "Versão Estática" se referem as informações de pré-processamento obtidas pela fase estática (sem etapa extendida) e as colunas "Versão Dinâmica" dizem respeito às informações atualizadas pela fase dinâmica (com etapa extendida).

A Tabela 7 mostra o comparativo entre as variações quanto às quantidades de nós envolvidas na resolução das instâncias. As colunas "Iterações" e "Nós explorados" dizem respeito aos mesmos conceitos que na seção anterior. As colunas "Nós podados" informam o número de nós descartados segundo o critério de limites e a coluna "Nós evitados" contabiliza o número de nós podados pelo critério de Violação Antecipada.

Perceba que o número de nós podados na variação com ambos os critérios é menor que o da outra variação, assim como esperado, pois o critério de poda por violação antecipada se propõe a eliminar os nós que levariam a soluções de custo superior à melhor solução conhecida ou mesmo que levariam a subproblemas inviáveis, que seriam exatamente podados pelo outro critério. Dessa forma, conseguimos impactar no número de iterações necessárias para satisfazer os critérios de parada do esquema de uma maneira geral, obtendo excelentes ganhos em instâncias como 21.2 e 21.3, e outros bastante expressivos, como nas instâncias 24, 29 e 37, onde reduzimos as iterações em torno da metade, se observarmos a versão estática desse critério de poda.

Tabela 7 – Comparativo entre os critérios de poda do esquema de *B&B* em (P) quanto à manipulação de nós do *B&B*.

Instância	Poda por Limites			Poda por Limites + Violação Antecipada							
				Versão Estática				Versão Dinâmica			
	Iterações	Nós Explorados	Nós Podados	Iterações	Nós Explorados	Nós Podados	Nós Evitados	Iterações	Nós Explorados	Nós Podados	Nós Evitados
10.1	57	199	49	51	129	36	80	4	22	3	13
10.2	1	21	1	2	19	1	6	0	1	1	0
10.3	17	104	8	16	51	6	66	0	1	1	0
10.4	282	1428	331	147	350	99	761	28	84	15	203
10.5	25	97	25	25	71	19	36	0	1	1	0
12.1	80	324	62	56	157	38	131	45	125	29	113
12.2	206	751	128	177	424	90	329	172	390	30	339
12.3	218	755	123	166	400	93	282	162	367	9	295
12.4	93	507	33	67	187	14	286	35	112	4	174
12.5	111	503	105	85	208	64	270	76	196	60	294
16.1	2659	11293	1075	1257	2836	464	3521	1198	2662	46	3428
16.2	813	4244	1318	372	1134	338	1650	320	941	310	1610
16.3	1079	4325	977	677	1667	459	1601	649	1585	142	1532
16.4	818	4123	1833	484	1353	667	1334	384	1183	574	1407
16.5	691	3354	1348	260	1042	658	1109	245	949	62	1103
21.1	2129	9527	2589	1002	2622	1060	2838	1010	2605	2	2949
21.2	234530	1254310	244889	62722	123572	28353	325761	29640	51618	9071	165787
21.3	301060	1412440	130261	15093	25900	2486	70580	10771	17621	1598	51910
21.4	10469	49629	13755	4496	10217	2656	15150	2540	5509	1273	10083
21.5	222	897	469	209	636	376	218	206	616	25	217
24	420178	2028133	59228	217379	497454	27508	833797	177753	391656	6266	709774
29	413338	2197653	325419	268459	629416	152136	1326631	301947	694640	153844	1553580
33	–	–	–	724211	1303616	0	7957494	3804767	6830836	1135098	39992197
37	–	–	–	570405	1519022	0	1882924	–	–	–	–
45	–	–	–	633182	1954240	0	9419686	–	–	–	–

Fonte: própria autoria

Para a versão dinâmica, conseguimos reduzir ainda mais o número de iterações necessárias para a resolução das instâncias, à exceção apenas das instâncias 21.1, 29 e 33. Para a instância 21.1, a quantidade ligeiramente superior se deve ao fato de a ordem entre os nós explorados pelo algoritmo de *B&B* ser diferente entre as versões estática e dinâmica, dado o menor número de variáveis no modelo obtido pela remoção de mais arcos nesta última. Para as instâncias 29 e 33 temos que a substituição da estrutura de lista de nós ativos da árvore de *B&B* possibilitou o encontro da solução ótima, o que não ocorria anteriormente, como podemos ver na Tabela 8, por isso o maior número de iterações. Para as instâncias 10.2, 10.3 e 10.5 o problema foi resolvido já na raiz da árvore de *Branch-and-Bound* graças às novas remoções de arcos realizadas pela versão dinâmica.

Na Tabela 8 exibimos os resultados quanto às soluções encontradas por essas variações. As colunas exibidas contém informações de mesma natureza das exibidas na Tabela 6. Entradas nas colunas "UB" com o valor ∞ indicam que nenhuma solução viável foi encontrada durante a busca pela solução ótima, e podem ocorrer em instâncias para as quais rejeitamos os nós da árvore de *B&B* que levariam à solução ótima. Para a coluna "*Status*" da versão dinâmica da poda por violação antevista, *EAB* significa que o critério de parada pelo erro absoluto entre as cotas foi alcançado e *MEM* significa que não obtivemos a solução ótima para a instância por conta da falta de memória disponível para continuar a execução do algoritmo.

Perceba que a redução no número de iterações obtida pela adição do critério de poda por violação antecipada nos permitiu diminuir consideravelmente o tempo necessário para a resolução de algumas instâncias, como no caso das 21.2, 21.3 e 29. Note que à exceção da instância 21.5, sempre obtivemos melhora no tempo de resolução, apesar de que nesse caso a diferença é irrisória.

Além disso, para a versão estática do critério de poda, conseguimos alcançar um critério de parada para as instâncias 33, 37 e 45 dentro do tempo limite de 24h, apesar de não termos obtido as soluções ótimas, o que não ocorreu para nenhuma das variações analisadas anteriormente.

Quanto à versão dinâmica, perceba a redução no tempo de execução do algoritmo de *Branch-and-Bound* obtida pela redução no número de iterações do algoritmo. Para as instâncias 29 e 33 fomos capazes de obter a solução ótima antes mesmo do tempo necessário para a versão estática, onde a solução havia sido perdida em um dos nós rejeitados.

8.6 CPLEX

Nesta seção, comparamos os resultados de tempo obtidos pela melhor estratégia desenvolvida com os tempos da ferramenta comercial CPLEX. Como podemos concluir ao analisarmos os resultados das seções anteriores, a melhor estratégia obtida foi o esquema de *Branch-and-Bound* em (P) onde consideramos a estratégia de ramificação por dicotomia usando limites superiores generalizados e ambos os critérios de poda descritos, junto à etapa de extensão da fase de pré-processamento. A Tabela 9 exibe o comparativo desses resultados, onde a coluna

Tabela 8 – Comparativo entre os critérios de poda do esquema de *B&B* em (P) quanto às soluções obtidas.

Instância	D	Poda por Limites				Poda por Limites e por Violação Antecipada					
						Versão Estática				Versão Dinâmica	
		Tempo CPU	UB	LB	*Status*	Tempo CPU	UB	LB	*Status*	Tempo CPU	*Status*
10.1	38	0s258ms	350	350	OPT	0s202ms	350	350	OPT	021ms	OPT
10.2	38	0s007ms	101	101	ITER	0s007ms	101	101	ITER	003ms	OPT
10.3	52	0s045ms	188	188	ITER	0s034ms	188	188	ITER	009ms	OPT
10.4	39	1s106ms	189	189	OPT	0s360ms	189	189	OPT	052ms	ITER
10.5	48	0s703ms	171	171	ITER	0s064ms	171	171	ITER	010ms	OPT
12.1	39	0s312	441	441	ITER	0s200ms	441	441	ITER	141ms	ITER
12.2	57	0s844	192	192	ITER	0s621ms	192	192	ITER	441ms	OPT
12.3	52	0s889	182	182	OPT	0s608ms	182	182	ITER	460ms	OPT
12.4	58	0s428	231	231	ITER	0s248ms	231	231	ITER	097ms	ITER
12.5	63	0s424	242	242	OPT	0s265ms	242	242	OPT	212ms	OPT
16.1	51	23s486	662	662	OPT	9s207ms	662	662	ITER	7s066ms	OPT
16.2	108	8s938	373	373	ITER	3s528ms	373	373	ITER	2s065ms	ITER
16.3	99	10s636	394	394	OPT	6s372ms	394	394	OPT	9s127ms	OPT
16.4	67	14s063	320	320	OPT	6s325ms	320	320	OPT	3s243ms	OPT
16.5	90	10s865	311	311	OPT	4s183ms	311	311	ITER	6s828ms	OPT
21.1	207	27s761	298	298	OPT	12s146ms	298	298	OPT	14s594ms	EAB
21.2	78	1h2min34s576	398	317,5	REJ	11min53s952ms	395	395	OPT	3min31s316ms	EAB
21.3	94	52min22s175	541	499,571	REJ	2min2s376ms	541	541	OPT	59s888ms	EAB
21.4	109	3min5s542	453	453	OPT	1min8s434ms	453	453	OPT	24s961ms	EAB
21.5	103	4s675	331	331	ITER	4s799ms	331	331	ITER	4s792ms	OPT
24	115	3h21min14s734	896	704,5	REJ	1h31min54s627ms	896	896	OPT	52min56s265ms	EAB
29	138	7h25min41s138ms	978	575,667	REJ	2h28min33s854ms	986	810,683	REJ	2h17min15s090ms	EAB
33	165	TLE	–	–	–	4h41min16s874ms	∞	879	REJ	16h47min53s957ms	EAB
37	176	TLE	–	–	–	12h16min43s426ms	∞	1018,95	REJ	–	MEM
45	613	TLE	–	–	–	10h35min11s756ms	∞	641,381	REJ	–	MEM

Fonte: própria autoria

"Tempos CMRPAM" representa os tempos da nossa melhor estratégia e a coluna "Tempos CPLEX" representa os tempos obtidos pelo CPLEX. Marcamos com $\star$ as instâncias para as quais não conseguimos encontrar a solução ótima.

Tabela 9 – Comparativo entre os resultados da melhor estratégia desenvolvida e do CPLEX.

	Instância	Tempos CMRPAM	Tempos CPLEX
	10.1	021ms	0s573ms
	10.2	003ms	0s131ms
	10.3	009ms	0s536ms
	10.4	052ms	1s161ms
	10.5	010ms	0s422ms
	12.1	141ms	2s618ms
	12.2	441ms	1s038ms
	12.3	460ms	2s6960ms
	12.4	097ms	3s584ms
	12.5	212ms	1s634ms
	16.1	7s066ms	17s598ms
	16.2	2s065ms	7s755ms
	16.3	9s127ms	10s139ms
	16.4	3s243ms	3s329ms
	16.5	6s828ms	6s769ms
	21.1	14s594ms	2min3s991ms
	21.2	3min31s316ms	1min41s004ms
	21.3	59s888ms	2mins31s201ms
	21.4	24s961ms	40s795
	21.5	4s792ms	16s688ms
	24	52min56s265ms	20min50s436ms
	29	2h17min15s090ms	26min49s402ms
	33	16h47min53s957ms	27min5s893ms
$\star$	37	MEM	4h13min40s500ms
$\star$	45	MEM	TLE

Fonte: própria autoria

Note que, excetuando-se a instância 21.2, conseguimos nos equiparar à ferramenta, ou mesmo superá-la por boa margem, nas instâncias com malhas de dimensão 10, 12, 16 e 21. Para as demais, o CPLEX consegue resolver otimamente as instâncias em tempos bastante inferiores aos nossos, reduzindo o tempo em mais da metade no caso da instância 24. O único caso em que essa ferramenta não consegue otimizar a instância dentro do tempo limite é na instância 45.

Os experimentos computacionais evidenciam que nossas estratégias de pré-processamento, ramificação e poda de nós no esquema de *Branch-and-Bound* são competitivas com as técnicas empregadas pelo CPLEX, incluindo planos de corte, para instâncias de ordem até 21. Entretanto, para as demais instâncias o CPLEX é bem mais eficiente. É provável que com o uso de planos de corte possamos melhorar a performance do nosso algoritmo.

9 CONSIDERAÇÕES E TRABALHOS FUTUROS

Neste trabalho investigamos o problema do *Caminho Mínimo com Restrição Probabilística de Atraso Máximo*, apresentando sua formulação matemática estocástica e sua equivalente formulação determinística. Ao máximo de nosso conhecimento, não parece existir trabalho na literatura que considere o parâmetro tempo sobre a mesma ótica do CMRPAM.

A partir da formulação determinística obtida, desenvolvemos uma decomposição baseada no modelo mestre-escravo de Benders, que utilizamos em um esquema de enumeração de partições do problema como um seletor inteligente. Os testes realizados mostraram que esse esquema carece de melhores cotas inferiores de forma a determinar a solução do problema proposto, consumindo muito tempo até mesmo para as menores instâncias testadas. Dessa forma, consideramos uma nova relaxação do problema (P) de CMRPAM, que resolvemos através do método do subgradiente, com o intuito de obter cotas inferiores mais fortes e, consequentemente, reduzir o tempo de resolução das instâncias. Entretanto, as cotas obtidas através desse método não se mostraram muito melhores que as de que já dispúnhamos.

Desenvolvemos, então, um esquema de *B&B* de forma a resolver diretamente o problema (P) em sua formulação determinística. Realizamos um pré-processamento no grafo de entrada de forma a remover arcos que certamente não fazem parte das soluções ótimas desejadas. Consideramos uma variação na estratégia de ramificação de forma a considerar apenas variáveis intencionando a construção iterativa do caminho desejado. Com isso, passamos a ramificar os nós através de dicotomia por *limite superior generalizado*. Finalmente, evitamos a ramificação de nós infrutíferos através da determinação de ramos da árvore de *B&B* que não contém nenhuma solução ótima aproveitando informações adquiridas na fase de pré-processamento.

Realizamos comparativos de tempo de solução dos nossos métodos com a ferramenta CPLEX. Como podemos visualizar nos resultados obtidos, para todas as instâncias de até médio porte conseguimos um ganho no tempo de resolução quando comparamos com o tempo dessa ferramenta. Para as instâncias de maior porte, o CPLEX se mostra mais eficiente.

Tendo em mente que essa ferramenta comercial faz uso extensivo de planos de corte de forma a acelerar a obtenção de soluções ótimas para os problemas com que se depara, imaginamos que sua vantagem reside nesse fato. Cremos que se considerássemos planos de corte semelhantes aos utilizados pelo CPLEX seríamos capazes de obter resultados melhores que este. Entretanto, decidimos não focar nessa questão em razão do limitado tempo de que dispomos para realizar e finalizar este trabalho.

Como trabalhos futuros, citamos:

Validações estatísticas Por tratar-se de um problema de natureza estocástica, validações estatísticas sobre as soluções obtidas devem ser realizadas de forma a garantir efetivamente a

conformidade com a margem de risco desejada para as rotas.

Número de cenários De posse de um ferramental estatístico para validar as soluções obtidas, pode-se tentar determinar limites inferiores para o número de cenários de atraso a considerar de forma a obter certas garantias de qualidade sobre as rotas a serem sugeridas.

Desigualdades Válidas Pouco se conhece sobre desigualdades válidas para o problema. Eventualmente, pretendemos investir nessa direção.

REFERÊNCIAS

ALEXOPOULOS, C. State space partitioning methods for stochastic shortest path problems. *Networks*, v. 30, n. 1, p. 9–21, 1997. ISSN 0028-3045.

ANDRADE, R. C. D.; ARARUNA, A. R.; LISSER, A. Probabilistic delay constrained shortest-path problem. In: *XIII ROADEF*. Angers: [s.n.], 2012. p. 127–128.

AVELLA, P.; BOCCIA, M.; SFORZA, A. A penalty function heuristic for the resource constrained shortest path problem. *European Journal of Operational Research*, v. 142, n. 2, p. 221–230, 2002. ISSN 0377-2217.

AVELLA, P.; BOCCIA, M.; SFORZA, A. Resource constrained shortest path problems in path planning for fleet management. *Journal of Mathematical Modelling and Algorithms*, v. 3, n. 1, p. 1–17, 2004. ISSN 15701166.

AVERBAKH, I. On the complexity of a class of combinatorial optimization problems with uncertainty. *Mathematical Programming*, v. 90, n. 2, p. 263–272, 2001. ISSN 0025-5610.

BARNHART, C.; BOLAND, N. L.; CLARKE, L. W.; JOHNSON, E. L.; NEMHAUSER, G. L.; SHENOI, R. G. Flight string models for aircraft fleeting and routing. *Transportation Science*, v. 32, n. 3, p. 208–220, 1998. ISSN 0041-1655.

BEASLEY, J. E.; CHRISTOFIDES, N. An algorithm for the resource constrained shortest path problem. *Networks*, v. 19, n. 4, p. 379–394, 1989. ISSN 00283045.

BELLMAN, R. On a routing problem. *Quarterly Applied Mathematics*, n. 16, p. 87–90, 1958.

BOLAND, N.; DETHRIDGE, J.; DUMITRESCU, I. Accelerated label setting algorithms for the elementary resource constrained shortest path problem. *Operations Research Letters*, v. 34, n. 1, p. 58–68, 2006. ISSN 01676377.

BONDY, J. A.; MURTY, U. S. *Graph Theory*. 1. ed. [S.l.]: Springer, 2008. ISBN 978-1-84628-969-9.

CARLYLE, W. M.; ROYSET, J. O.; WOOD, R. K. Lagrangian relaxation and enumeration for solving constrained shortest-path problems. *Networks*, Wiley-Interscience, New York, NY, USA, v. 52, n. 4, p. 256–270, 2008. ISSN 0028-3045.

CHEN, S.; NAHRSTEDT, K. On finding multi-constrained paths. In: *ICC '98. 1998 IEEE International Conference on Communications. Conference Record.* [S.l.]: IEEE, 1998. v. 2, p. 874–879. ISBN 0-7803-4788-9.

CHENG, J.; KOSUCH, S.; LISSER, A. Stochastic shortest path problem with uncertain delays. In: LUZ, C. J.; VALENTE, F. (Ed.). *Proceedings of the 1st International Conference on Operations Research and Enterprise Systems (ICORES-2012)*. [S.l.: s.n.], 2012. p. 256—-264.

DANTZIG, G. B. Discrete-variable extremum problems. *Operations Research*, n. 5, p. 266–277, 1957.

DANTZIG, G. B. On the shortest route through a network. *Management Science*, n. 6, p. 187–190, 1960.

DESROCHERS, M.; DESROSIERS, J.; SOLOMON, M. M. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, v. 40, n. 2, p. 342–354, 1992. ISSN 0030-364X.

DESROCHERS, M.; SOUMIS, F. The shortest-path with scheduling constraints. *INFOR*, v. 26, n. 3, p. 191–212, 1988.

DESROSIERS, J.; DUMAS, Y.; SOLOMON, M. M.; SOUMIS, F. Time constrained routing and scheduling. In: BALL, M. O.; MAGNANTI, T. L.; MONMA, C. L.; NEMHAUSER, G. L. (Ed.). *Handbook in Operations Research and Management Science 8*. North-Holland, Amsterdam: [s.n.], 1995. cap. 2, p. 35–139.

DIJKSTRA, E. W. A note on two problems in connexion with graphs. *Numerische Mathematik*, n. 1, p. 169–271, 1959.

DUMITRESCU, I.; BOLAND, N. Improved preprocessing, labeling and scaling algorithms for the weight-constrained shortest path problem. *Networks*, v. 42, n. 3, p. 135–153, 2003. ISSN 0028-3045.

ELIMAM, A. A.; KOHLER, D. Two engineering applications of a constrained shortest-path model. *European Journal of Operational Research*, v. 103, n. 3, p. 426–438, 1997. ISSN 03772217.

ESPINOZA, D.; GARCIA, R.; GOYCOOLEA, M.; NEMHAUSER, G. L.; SAVELSBERGH, M. W. P. Per-seat, on-demand air transportation Part I: Problem description and an integer multicommodity flow model. *Transportation Science*, v. 42, n. 3, p. 263–278, 2008. ISSN 0041-1655.

FORD JR., L. R. *Network Flow Theory*. Santa Monica, California, 1956.

FREITAS, A. T. *Árvore de Subgradiente com Pré-Fase VNS-Lagrangeana para a Árvore Geradora com Restrição de Grau Máximo nos Vértices*. Dissertação (Mestrado) — Universidade Federal do Ceará, 2011.

GABREL, V.; MURAT, C. Robust shortest path problems. p. 71–93, 2007.

GABREL, V.; MURAT, C.; WU, L. New models for the robust shortest path problem: complexity, resolution and generalization. *Annals of Operations Research*, 2011. ISSN 0254-5330.

GAREY, M. R.; JOHNSON, D. S. *Computers and intractability: A Guide to the Theory of NP-Completeness*. San Francisco: Freeman, 1979. 174 p. ISBN 0716710447.

GUO, L.; MATTA, I. Search space reduction in QoS routing. In: *Proceedings. 19th IEEE International Conference on Distributed Computing Systems (Cat. No.99CB37003)*. [S.l.]: IEEE Comput. Soc, 1999. p. 142–149. ISBN 0-7695-0222-9.

HALPERN, J.; PRIESS, I. Shortest path with time constraints on movement and parking. *Networks*, v. 4, n. 3, p. 241–253, 1974. ISSN 00283045.

HANDLER, G. Y.; ZANG, I. A dual algorithm for the constrained shortest path problem. *Networks*, Wiley Subscription Services, Inc., A Wiley Company, v. 10, n. 4, p. 293–309, 1980. ISSN 00283045.

HASSIN, R. Approximation schemes for the restricted shortest path problem. *Mathematics of Operations Research*, v. 17, n. 1, p. 36–42, 1992. ISSN 0364-765X.

HERNANDES, F. *Algoritmos para problemas de grafos com incertezas*. 142 p. Tese (PhD) — Universidade Estadual de Campinas, 2007.

HIRSCH, M. J.; PARDALOS, P. M.; MURPHEY, R.; GRUNDEL, D. Advances in cooperative control and optimization. In: *7th International Conference on Cooperative Control and Optimization*. [S.l.]: Springer-Verlag Berlin / Heidelberg, 2007. p. 422.

IRNICH, S.; DESAULNIERS, G. Shortest path problems with resource constraints. In: DESAULNIERS, G.; DESROSIERS, J.; SOLOMON, M. M. (Ed.). *Column Generation*. [S.l.]: Springer US, 2005. cap. 2, p. 33–65. ISBN 978-0-387-25485-2.

JAFFE, J. M. Algorithms for finding paths with multiple constraints. *Networks*, Wiley Subscription Services, Inc., A Wiley Company, v. 14, n. 1, p. 95–116, 1984. ISSN 00283045.

JAUMARD, B.; SEMET, F.; VOVOR, T. G. A two-phase resource constrained shortest path algorithm for acyclic graphs. *Les Cahiers du GERARD*, p. 46, 1999.

JI, X. Models and algorithm for stochastic shortest path problem. *Applied Mathematics and Computation*, v. 170, n. 1, p. 503–514, 2005. ISSN 00963003.

JIMÉNEZ, V. M.; MARZAL, A. Computing the k shortest paths: A new algorithm and an experimental comparison. In: VITTER, J. S.; ZAROLIAGIS, C. D. (Ed.). *3rd Workshop on Algorithm Engineering*. [S.l.]: Springer Berlin Heidelberg, 1999. p. 15–29.

JOKSCH, H. C. The shortest route problem with constraints. *Journal of Mathematical Analysis and Applications*, v. 14, n. 2, p. 191–197, 1966. ISSN 0022247X.

JUTTNER, A.; SZVIATOVSKI, B.; MECS, I.; RAJKO, Z. Lagrange relaxation based method for the QoS routing problem. In: *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society*. [S.l.]: IEEE, 2001. v. 2, p. 859–868. ISBN 0-7803-7016-3.

KARASAN, O. E.; PINAR, M. c.; YAMAN, H. The robust shortest path problem with interval data. *Computers & Operations Research*, 2001.

KOHL, N. *Exact methods for time constrained routing and related scheduling problems*. 234 p. Tese (PhD) — Technical University of Denmark, 1995.

KORKMAZ, T.; KRUNZ, M. Multi-constrained optimal path selection. In: *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society*. [S.l.]: IEEE, 2001. v. 2, p. 834–843. ISBN 0-7803-7016-3.

KORKMAZ, T.; KRUNZ, M. A randomized algorithm for finding a path subject to multiple QoS requirements. *Computer Networks*, v. 36, n. 2-3, p. 251–268, 2001. ISSN 13891286.

KOSUCH, S.; LISSER, A. Stochastic shortest path problem with delay excess penalty. *Electronic Notes in Discrete Mathematics*, v. 36, p. 511–518, 2010. ISSN 15710653.

LORENZ, D. H.; ORDA, A.; RAZ, D.; SHAVITT, Y. Efficient QoS partition and routing of unicast and multicast. In: *2000 Eighth International Workshop on Quality of Service. IWQoS 2000*. [S.l.]: IEEE, 2000. p. 75–83. ISBN 0-7803-6266-7.

LORENZ, D. H.; RAZ, D. A simple efficient approximation scheme for the restricted shortest path problem. *Operations Research Letters*, v. 28, n. 5, p. 213–219, 2001. ISSN 01676377.

MEHLHORN, K.; ZIEGELMANN, M. Resource constrained shortest paths. In: PATERSON, M. S. (Ed.). *Algorithms - ESA 2000*. [S.l.]: Springer Berlin / Heidelberg, 2000, (Lecture Notes in Computer Science, v. 1879). p. 326–337. ISBN 978-3-540-41004-1.

MIEGHEM, P. V.; KUIPERS, F. A.; KORKMAZ, T.; KRUNZ, M.; CURADO, M.; MONTEIRO, E.; MASIP-BRUIN, X.; SOLÉ-PARETA, J.; SÁNCHEZ-LÓPEZ, S. Quality of Service routing. In: SMIRNOV, M. (Ed.). *Quality of Future Internet Services*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003. (Lecture Notes in Computer Science, v. 2856), p. 80–117. ISBN 978-3-540-20193-9.

MIEGHEM, P. V.; NEVE, H. D.; KUIPERS, F. Hop-by-hop quality of service routing. *Computer Networks*, v. 37, n. 3-4, p. 407–423, 2001. ISSN 13891286.

MINTY, G. J. A comment on the shortest-route problem. *Operations Research*, n. 5, p. 724, 1957.

MINTY, G. J. A variant on the shortest-route problem. *Operations Research*, n. 6, p. 882–883, 1958.

MIRCHANDANI, P. B. Shortest distance and reliability of probabilistic networks. *Computers & Operations Research*, v. 3, n. 4, p. 347–355, 1976. ISSN 03050548.

MONTEMANNI, R.; GAMBARDELLA, L. An exact algorithm for the robust shortest path problem with interval data. *Computers & Operations Research*, v. 31, n. 10, p. 1667–1680, 2004. ISSN 03050548.

MONTEMANNI, R.; GAMBARDELLA, L. *Robust shortest path problems with uncertain costs*. Manno, Switzerland, 2008. 11 p.

MONTEMANNI, R.; GAMBARDELLA, L.; DONATI, A. A branch and bound algorithm for the robust shortest path problem with interval data. *Operations Research Letters*, v. 32, n. 3, p. 225–232, 2004. ISSN 01676377.

MONTEMANNI, R.; GAMBARDELLA, L. M. The robust shortest path problem with interval data via Benders decomposition. *4OR: A Quarterly Journal of Operations Research*, Springer Berlin / Heidelberg, v. 3, n. 4, p. 315–328, 2005. ISSN 1619-4500.

MUHANDIRAMGE, R.; BOLAND, N. Simultaneous solution of Lagrangean dual problems interleaved with preprocessing for the weight constrained shortest path problem. *Networks*, v. 53, n. 4, p. 358–381, 2009. ISSN 00283045.

NEVE, H. D.; MIEGHEM, P. V. TAMCRA: a tunable accuracy multiple constraints routing algorithm. *Computer Communications*, v. 23, n. 7, p. 667–679, 2000. ISSN 01403664.

NYGAARD, R. *Shortest path methods in representation and compression of signals and image contours*. 168 p. Tese (PhD) — Norwegian University of Science and Technology, Faculty of Information Technology, Mathematics and Electrical Engineering, 2000.

NYGAARD, R.; MELNIKOV, G.; KATSAGGELOS, A. K. A rate distortion optimal ECG coding algorithm. *IEEE transactions on bio-medical engineering*, v. 48, n. 1, p. 28–40, 2001. ISSN 0018-9294.

ORDA, A. Routing with end-to-end QoS guarantees in broadband networks. *IEEE/ACM Transactions on Networking*, v. 7, n. 3, p. 365–374, 1999. ISSN 10636692.

PHILLIPS, C. A. The network inhibition problem. In: *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing - STOC '93*. New York, New York, USA: ACM Press, 1993. p. 776–785. ISBN 0897915917.

POLYCHRONOPOULOS, G. H.; TSITSIKLIS, J. N. Stochastic shortest path problems with recourse. *Networks*, v. 27, n. 2, p. 133–143, 1996. ISSN 0028-3045.

PUGLIESE, L. d. P.; GUERRIERO, F. A computational study of solution approaches for the resource constrained elementary shortest path problem. *Annals of Operations Research*, v. 201, n. 1, p. 131–157, 2012. ISSN 0254-5330.

PUGLIESE, L. d. P.; GUERRIERO, F. A reference point approach for the resource constrained shortest path problems. *Transportation Science*, 2012. ISSN 0041-1655.

REEVES, D.; SALAMA, H. A distributed algorithm for delay-constrained unicast routing. *IEEE/ACM Transactions on Networking*, v. 8, n. 2, p. 239–250, 2000. ISSN 10636692.

RIGHINI, G.; SALANI, M. New dynamic programming algorithms for the resource constrained elementary shortest path problem. *Networks*, v. 51, n. 3, p. 155–170, 2008. ISSN 00283045.

ROSS, S. M. *A first course in Probability*. [S.l.]: Prentice Hall, 2009. 552 p. p. ISBN 978-0136033134.

ROY, B. Robustness in operational research and decision aiding: A multi-faceted issue. *European Journal of Operational Research*, v. 200, n. 3, p. 629–638, 2010. ISSN 03772217.

ROYSET, J. O. Routing military aircraft with a constrained shortest-path algorithm. *Military Operations Research*, v. 14, n. 3, 2009. ISSN 10825983.

SCHRIJVER, A. On the history of the shortest path problem. *Documenta Mathematica*, ISMP, p. 155–167, 2012.

SIGAL, C. E.; PRITSKER, A. A. B.; SOLBERG, J. J. The stochastic shortest route problem. *Operations Research*, v. 28, n. 5, p. 1122–1129, 1980. ISSN 0030-364X.

SKISCIM, C. C.; GOLDEN, B. L. Solving k-shortest and constrained shortest path problems efficiently. *Annals of Operations Research*, v. 20, n. 1, p. 249–282, 1989. ISSN 0254-5330.

SRIRAM, R.; MANIMARAN, G.; MURTHY, C. S. R. Preferred link based delay-constrained least-cost routing in wide area networks. *Computer Communications*, v. 21, n. 18, p. 1655–1669, 1998. ISSN 01403664.

TREVIZAN, F. W.; VELOSO, M. M. Finding objects through stochastic shortest path problems. In: *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2013. (AAMAS '13), p. 547–554. ISBN 978-1-4503-1993-5.

VERWEIJ, B.; AHMED, S.; KLEYWEGT, A. J.; NEMHAUSER, G. L.; SHAPIRO, A. The sample average approximation method applied to stochastic routing problems: A computational study. *Computational Optimization and Applications*, Springer Netherlands, v. 24, n. 2, p. 289–333, 2003. ISSN 0926-6003.

WIDYONO, R. *The design and evaluation of routing algorithms for real-time channels*. Berkeley, 1994.

WIERZBICKI, A. P.; MAKOWSKI, M.; WESSELS, J. *Model-based Decision Support Methodology with Environmental Applications*. Dordrecht, The Netherlands: Kluwer Academic Publishers, 2000. 492 p. ISBN 0792363272.

XUE, G. Primal-dual algorithms for computing weight-constrained shortest paths and weight-constrained minimum spanning trees. In: *2000 IEEE International Performance, Computing, and Communications Conference*. [S.l.]: IEEE, 2000. p. 271–277. ISBN 0-7803-5979-8.

YU, G.; YANG, J. On the robust shortest path problem. *Computers & Operations Research*, v. 25, n. 6, p. 457–468, 1998. ISSN 0305-0548.

YUAN, X. Heuristic algorithms for multiconstrained quality-of-service routing. *IEEE/ACM Transactions on Networking*, v. 10, p. 244–256, 2002.

ZABARANKIN, M.; URYASEV, S.; PARDALOS, P. M. Optimal risk path algorithms. In: MURPHEY, R.; PARDALOS, P. (Ed.). *Cooperative Control, and Optimization*. Kluwer, Dordrecht, The Netherlands: Springer, 2002. p. 273–298.

ZIEGELMANN, M. Constrained shortest paths and related problems - constrained network optimization. VDM Verlag, 2007.

ZIELIŃSKI, P. The computational complexity of the relative robust shortest path problem with interval data. *European Journal of Operational Research*, v. 158, n. 3, p. 570–576, 2004. ISSN 03772217.

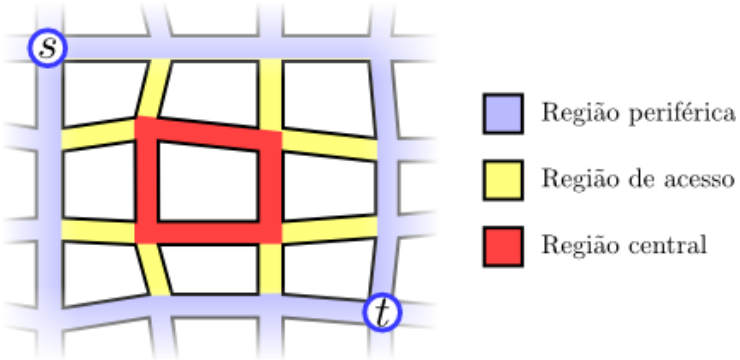
APÊNDICE A – INSTÂNCIAS DE TESTE

Todas as instâncias de teste reportadas neste trabalho constituem-se de grafos simples e direcionados do tipo malha[1] quadrada completa, com custos das arestas escolhidos aleatoriamente de maneira uniforme. Como dito no Capítulo 3, os atrasos das arestas são regidos por variáveis aleatórias de distribuição de valores discretos. Para nossos testes, consideramos a distribuição de Poisson, embora as técnicas de resolução que propomos sejam independentes da distribuição dos atrasos. A seguir, explicamos o conceito de *zoneamento da malha viária*, que influencia na forma como escolhemos os custos e as médias das variáveis aleatórias associadas às arestas nas nossas instâncias.

A.1 Zoneamento da malha viária

Para a definição dos atrasos de nossas instâncias, partimos do princípio de que as malhas viárias que elas representam possuem *zonas de atraso diferenciadas*. No caso de malhas urbanas, esse zoneamento reflete as diferenças de tempo que levamos em média para percorrer vias situadas na região periférica, na região de acesso e na região central da zona urbana. Logicamente, esperamos que o tráfego nas vias da região periférica seja menos intenso que na região de acesso e muito menos intenso que na região central, o que nos permite concluir que, em média, os tempos de percursos na primeira região sejam os menores, seguidos dos tempos na segunda e terceira, respectivamente. A Figura 12 ilustra um zoneamento em três regiões da malha viária apresentada no Capítulo 2.

Figura 12 – Exemplo de zoneamento em três regiões.



Fonte: própria autoria

É a partir desse princípio que definimos as *classes de atraso*, que nada mais são que conjuntos de todas as arestas relacionadas a vias em uma mesma zona. Arestas em uma mesma

[1] N.A.: do inglês: *grid*

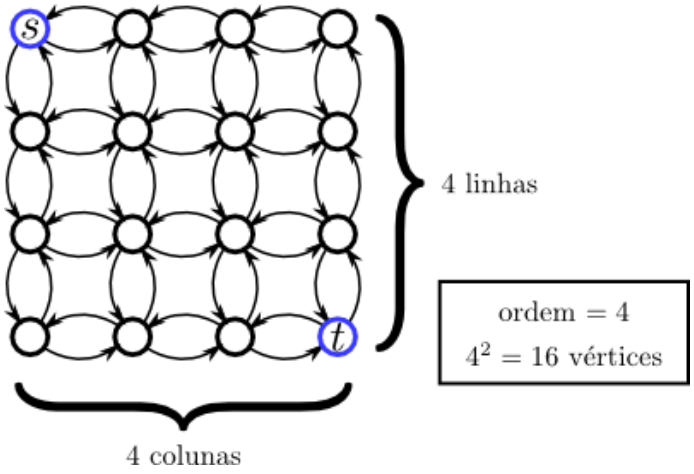
classe de atraso possuem valores de atraso regidos por variáveis aleatórias independentes de mesma média. Nos nossos testes computacionais, usamos instâncias de três a cinco classes distintas de atrasos.

A.2 Parâmetros de definição

Nossas instâncias foram geradas por um programa desenvolvido para esse fim, ao qual informamos os diversos parâmetros necessários para a determinação de uma instância e que gera um formato de saída lido pelo programa que criamos para realizar os testes computacionais reportados no Capítulo 8. Descrevemos a seguir os parâmetros necessários e seus significados na construção de cada instância.

O número de vértices em cada instância é determinado pelo que chamamos de *ordem* da malha, que consiste na quantidade de "linhas" ou "colunas" de vértices que obtemos quando dispomos o grafo em um plano. O quadrado da ordem em uma instância é exatamente o seu número de vértices. A Figura 13 ilustra o conceito para a instância de exemplo dada no Capítulo 2.

Figura 13 – Ilustração do conceito de *ordem* de uma instância.



Fonte: própria autoria

O número de classes de atraso é informado e, com essa informação, passamos à divisão das arestas nessas classes, dada por camadas de maneira visualmente semelhante ao exemplo na Figura 12. Informamos quantas linhas e colunas pertencem a cada classe da mais externa à mais interna. A partir dessa divisão, informamos as médias das variáveis aleatórias e o intervalo de valoração dos custos das arestas de cada classe, que serão escolhidos de maneira uniforme para a definição da instância. Lembramos que os valores efetivos de atraso de cada aresta em cada cenário de atraso são gerados aleatoriamente durante a busca pela solução a partir dos valores de média informados.

O número de cenários a simular também é informado e constitui parte da instância. Não informamos valores inferiores a 30, de forma a obtermos soluções aceitáveis e expressivas do ponto de vista estatístico. Nos nossos testes, geramos instâncias com 50 cenários.

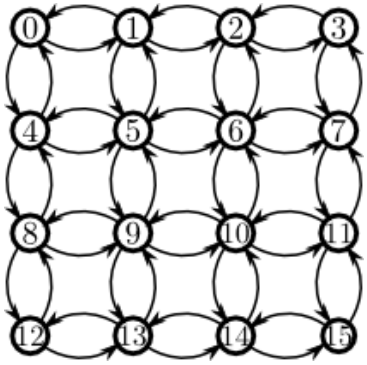
Por fim, informamos os vértices origem e destino do caminho e inserimos o valor do fator de risco e do atraso máximo desejado.

A.3 Tabela de definição de instâncias

Na Tabela 10 exibimos os parâmetros utilizados para definir todas as instâncias para os resultados reportados no Capítulo 8.

A primeira coluna informa a ordem da malha daquela instância. Geramos 5 instâncias distintas de cada uma das ordens 10, 12, 16 e 21. A diferenciação entre instâncias com malhas de mesma ordem se dá no valor apresentado na segunda coluna. Para as outras ordens, apenas uma instância foi gerada. A terceira coluna se refere ao número de classes de atraso da instância. A quarta e a quinta colunas informam respectivamente as médias de atraso de cada classe e o intervalo de valoração dos custos das arestas nessas classes. A sexta coluna indica quantas arestas constituem cada classe. Finalmente, a sétima coluna informa o vértice destino da instância. Os vértices são numerados de 0 a $|V| - 1$ a partir do vértice superior esquerdo quando desenhamos o grafo no plano, assim como o exemplo na Figura 14. Para todas as nossas instâncias, o vértice origem é o vértice de número 0.

Figura 14 – Exemplo de como os vértices de uma malha de ordem 4 são numerados.



Fonte: própria autoria

Tabela 10: Valores dos parâmetros para cada instância de teste gerada.

Ordem	Versão	Nº Classes	Médias de Atraso	Interv. de Custo	Nº Arestas	Destino
10	1	3	2	$[1;100]$	240	55
			4	$[1;50]$	112	
			8	$[1;30]$	8	
10	2	3	2	$[1;20]$	240	55
			4	$[1;40]$	112	
			8	$[1;80]$	8	
10	3	3	3	$[1;25]$	240	55
			7	$[1;45]$	112	
			10	$[1;85]$	8	
10	4	3	2	$[1;25]$	240	55
			5	$[1;45]$	112	
			7	$[1;85]$	8	
10	5	3	3	$[1;20]$	240	55
			5	$[1;40]$	112	
			8	$[1;80]$	8	
12	1	3	3	$[1;100]$	408	78
			4	$[1;50]$	112	
			8	$[1;30]$	8	
12	2	3	3	$[1;20]$	408	78
			6	$[1;40]$	112	
			9	$[1;80]$	8	
12	3	3	3	$[1;20]$	408	78
			5	$[1;40]$	112	
			8	$[1;80]$	8	
12	4	3	3	$[1;25]$	408	78
			7	$[1;45]$	112	
			10	$[1;90]$	8	
12	5	3	4	$[1;25]$	408	78
			6	$[1;45]$	112	
			8	$[1;90]$	8	
16	1	3	2	$[1;100]$	600	136
			4	$[1;50]$	352	
			8	$[1;30]$	8	
16	2	3	4	$[1;25]$	600	136
			7	$[1;55]$	312	
			10	$[1;95]$	48	
16	3	3	4	$[1;30]$	600	136
			6	$[1;60]$	312	
			8	$[1;90]$	48	
16	4	3	2	$[1;30]$	600	136
			5	$[1;60]$	312	
			9	$[1;90]$	48	

Fonte: própria autoria

Tabela 10: (continuação)

Ordem	Versão	Nº Classes	Médias de Atraso	Interv. de Custo	Nº Arestas	Destino
16	5	3	3	$[1;25]$	600	136
			6	$[1;55]$	312	
			9	$[1;90]$	48	
21	1	3	8	$[1;30]$	1240	242
			10	$[1;35]$	416	
			15	$[1;50]$	24	
21	2	4	2	$[1;20]$	1056	242
			4	$[1;45]$	456	
			6	$[1;70]$	144	
			8	$[1;90]$	24	
21	3	4	3	$[1;20]$	1056	242
			4	$[1;45]$	456	
			7	$[1;70]$	144	
			8	$[1;90]$	24	
21	4	4	3	$[1;30]$	1056	242
			5	$[1;50]$	456	
			9	$[1;70]$	144	
			11	$[1;90]$	24	
21	5	4	3	$[1;20]$	1056	242
			4	$[1;40]$	456	
			8	$[1;60]$	144	
			11	$[1;80]$	24	
24	-	3	3	$[1;50]$	1680	325
			6	$[1;75]$	480	
			9	$[1;100]$	48	
29	-	3	3	$[1;50]$	2408	480
			6	$[1;75]$	760	
			9	$[1;100]$	80	
33	-	4	3	$[1;50]$	2200	612
			5	$[1;70]$	1400	
			7	$[1;90]$	544	
			9	$[1;120]$	80	
37	-	4	3	$[1;50]$	3304	760
			5	$[1;70]$	1400	
			7	$[1;90]$	544	
			9	$[1;120]$	80	
45	-	3	10	$[1;30]$	5520	1196
			14	$[1;40]$	2112	
			18	$[1;50]$	288	

Fonte: própria autoria

APÊNDICE B – ESTIMAÇÃO DE PROBABILIDADE DOS CENÁRIOS

Como afirmamos ao longo do texto, linearizamos a restrição probabilística de limite de atraso através de uma amostra dos cenários de atraso das arestas. Entretanto, após a linearização dessa restrição, introduzimos novas restrições onde necessitamos indicar a probabilidade de ocorrência de cada cenário. Como não podemos precisar exatamente a probabilidade de um cenário escolhido dentre todos os possíveis, utilizamos estimativas desse valor. A seguir, exibimos como essa estimativa é calculada para os testes computacionais reportados.

Supomos que a probabilidade de cada cenário está relacionada à *frequência relativa* de ocorrência de valores de atraso entre as amostragens realizadas. A frequência relativa de um valor em uma lista de valores é definida como a razão entre o número de ocorrências desse valor e a quantidade de valores nessa lista.

Nos baseamos na seguinte propriedade de variáveis de Poisson:

Proposição 5. *Se X e Y são variáveis aleatórias independentes tais que $X \sim \text{Poisson}(\lambda)$ e $Y \sim \text{Poisson}(\pi)$, então a variável aleatória Z definida como $Z := X + Y$ é tal que $Z \sim \text{Poisson}(\lambda + \pi)$.*

Demonstração. Por definição, a PDF de Z será a convolução das PDFs de X e Y. Assim:

$$\begin{aligned}
\mathbb{P}(Z = w) &= \mathbb{P}(X + Y = w) && \text{(20a)}\\
&= \sum_{i=0}^{w} \mathbb{P}(X = i \wedge Y = w - i) && \text{(20b)}\\
&= \sum_{i=0}^{w} \mathbb{P}(X = i)\ \mathbb{P}(Y = w - i) && \text{(20c)}\\
&= \sum_{i=0}^{w} \left(\frac{\mathrm{e}^{-\lambda}\lambda^{i}}{i!}\right)\left(\frac{\mathrm{e}^{-\pi}\pi^{w-i}}{(w-i)!}\right) && \text{(20d)}\\
&= \frac{\mathrm{e}^{-(\lambda+\pi)}}{w!}\sum_{i=0}^{w}\frac{w!}{i!\,(w-i)!}\pi^{w-i}\lambda^{i} && \text{(20e)}\\
&= \frac{\mathrm{e}^{-(\lambda+\pi)}}{w!}\sum_{i=0}^{w}\binom{w}{i}\pi^{w-i}\lambda^{i} && \text{(20f)}\\
&= \frac{\mathrm{e}^{-(\lambda+\pi)}}{w!}(\lambda+\pi)^{w} && \text{(20g)}\\
&\therefore Z \sim \text{Poisson}(\lambda + \pi)
\end{aligned}$$

□

Assim, temos que as somas dos valores de atraso de todas as arestas em uma mesma classe de atraso[1] podem ser consideradas amostras de variáveis aleatórias de Poisson de média conhecida. Denominamos por s_k^c a soma dos valores de atraso das arestas da classe c obtidos no cenário k. Supomos, então, que a probabilidade de ocorrência de cada um desses valores de soma é, na realidade, sua frequência relativa aos cenários gerados, definindo portanto o evento de ocorrência de um cenário k como sendo a conjunção dos eventos de obtenção dos seus valores de soma para cada classe, normalizado de forma que o somatório das probabilidades para todos os cenários seja igual a 1.

Dessa forma, seja o vetor S_c, de dimensão $|K|$, tal que a k-ésima componente guarda o valor de s_k^c, e seja a função freq tal que $\mathrm{freq}(s_k^c, S_c)$ é a frequência relativa do valor s_k^c dentro do vetor S_c. Sendo C o conjunto de classes de atraso, temos que a probabilidade de um cenário $k \in K$ é estimada por:

$$\hat{\mathbb{P}}(k) = \prod_{c \in C} \mathrm{freq}(s_k^c, S_c) \tag{21}$$

Por fim, esse valor é normalizado de forma que $\sum_{k \in K} \hat{\mathbb{P}}(k) = 1$, dividindo-se a estimativa pelo somatório das estimativas de todos os cenários.

[1] A definição consta no Apêndice A