BENJAMIN MARIOTTI FELDMANN

DETERMINAÇÃO DE LINHAS DE TRANSPORTE NA OPERAÇÃO DE CARGA FRACIONADA

SÃO PAULO

2019

DETERMINAÇÃO DE LINHAS DE TRANSPORTE NA OPERAÇÃO DE CARGA FRACIONADA

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo, como requisito parcial para a obtenção do título de Mestre em Ciências

Orientador: Prof. Dr. André Bergsten Mendes

SÃO PAULO

2019

Autorizo a reprodução e divulgação total ou parcial deste trabalho, por qualquer meio convencional ou eletrônico, para fins de estudo e pesquisa, desde que citada a fonte.

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, _____ de _____ de _____

Assinatura do autor: _____

Assinatura do orientador: _____

Catalogação-na-publicação

Feldmann, Benjamin Mariotti
Determinação de linhas de transporte na operação de carga fracionada / B. M. Feldmann -- versão corr. -- São Paulo, 2019.
114 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Naval e Oceânica.

1.Carga fracionada 2.Linhas de transporte 3.Heurística 4.Caminhos mínimos com janelas de tempo I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Naval e Oceânica II.t.

AGRADECIMENTOS

Agradeço a todos os colegas de cursos e de trabalho que acompanharam toda essa jornada. Também agradeço à minha família, que conseguiu manter em níveis aceitáveis as perguntas sobre quando o projeto seria finalizado. A todos os amigos que me apoiaram e aos siblings, que me fizeram dar risadas quando não achei que fosse possível.

Um especial agradecimento ao Leandro Meili, por todo o incentivo, sugestões e compartilhamentos que sempre forneceu de maneira atenciosa.

Um agradecimento especial ao Vitor Marques, por toda a prontidão e energia no auxílio da coleta de dados. Também agradeço ao Dalton Vecchini e ao Omar Passos, sempre prestativos no envio de dados e resolução de dúvidas.

Agradeço ao meu orientador Prof. Dr. André Bergsten Mendes por todos os conselhos e compreensão, e ao Prof. Dr. Marco Antônio Brinati, pelo exemplo de vida.

RESUMO

FELDMANN, B. M. Determinação de linhas de transporte na operação de carga fracionada. 2019. 114 f. Tese (Mestrado) – Escola Politécnica, Universidade de São Paulo, São Paulo, 2017.

No presente trabalho, é abordada a operação de transporte de cargas fracionadas, especificamente a determinação de quais linhas de transporte deverão ser ofertadas dentro de uma rede de terminais, de maneira a atender toda a demanda no nível de serviço desejado ao menor custo possível. Para tanto, é feita inicialmente uma descrição do problema de transporte de carga fracionada, seguido de uma revisão bibliográfica de trabalhos anteriores que já tenham abordado o tema. É então realizada a delimitação do escopo do estudo e a proposição de um modelo matemático em programação linear inteira-mista. Em seguida, é apresentado um algoritmo de resolução, consistindo na aplicação de uma heurística construtiva e uma heurística de melhoria, ambas embasadas na aplicação de caminhos mínimos com janelas de tempo a partir de custos marginais. O método é delineado para três versões do problema, estipuladas a partir de diferentes tratamentos à restrição de caminhos em formato de árvore dentro do sistema. Primeiramente, o algoritmo é aplicado a pequenas instâncias fictícias, realizando a comparação com a modelagem em programação linear inteira-mista proposta. Na maioria dos casos, não houve diferença nos valores de função objetivo encontrados, embora tenham sido identificados *gaps* grandes no processamento. Posteriormente, é realizada a aplicação a dados reais de uma transportadora brasileira. Para as três versões do problema, a redução de custos potencial identificada é significativa, com tempos de processamento similares ou menores do que o encontrado na literatura. Por fim, os resultados obtidos são discutidos sendo apresentadas considerações finais acerca do trabalho realizado e possíveis melhorias para pesquisas futuras.

Palavras-chave: carga fracionada, linhas de transporte, heurística construtiva, melhorias locais, caminhos mínimos com janelas de tempo

ABSTRACT

FELDMANN, M. B. Heuristics for service network design of less-than-truckload transportation. 2019. 114 f. Tese (Mestrado) – Escola Politécnica, Universidade de São Paulo, São Paulo, 2017.

At the present work, the operation of less-than-truckload (LTL) will be studied, more specifically the determination of which lines will be offered in a network of terminals. The service network design must attend all demands, respecting their deadlines while aiming cost reductions. The objective of this work is to propose algorithms to solve the service network design problem of LTL operations, reducing operation costs while respecting specified service levels. First, a brief introduction to the problem is made, and similar research is reviewed. Then the scope of the research is determined and a mathematical model of the problem in mixed-integer programming is presented. Next, an algorithm is proposed, consisting in a constructive heuristic followed by a local search. Both phases are based on finding minimum paths with time windows using marginal costs along the network. Three different versions of the problem are analyzed, shifting the approach given to the constraint of in-tree structure that shipments should follow in the network. The algorithm is firstly tested to small fictional instances, allowing comparison to the mixed programming model proposed earlier. No relevant differences between objective functions were found, even though substantial gaps values were identified during processing. A second test used a real dataset of a Brazilian LTL carrier. In all versions of the problem the operation cost reduction was promising, with processing times similar to the ones found in literature. The conclusion provides a discussion of the obtained results and recommendations for future research.

Key words: less-than-truckload, service lines, constructive heuristics, local search, minimum paths with time windows

SUMÁRIO

LISTA DE TABELAS **5**
LISTA DE FIGURAS **6**
1 INTRODUÇÃO **10**
2 REVISÃO BIBLIOGRÁFICA **18**
3 DESCRIÇÃO DO PROBLEMA E MODELAGEM MATEMÁTICA **28**
3.1 MODELO SEM RESTRIÇÕES DE CAMINHOS 31
3.2 MODELO COM CAMINHOS FIXOS 36
3.3 MODELO COM CAMINHOS POR DIA DA SEMANA 37
4 MÉTODO DE SOLUÇÃO **39**
4.1 CAMINHOS MÍNIMOS COM JANELAS DE TEMPO 39
4.2 HEURÍSTICA CONSTRUTIVA 43
4.2.1 Algoritmo sem restrições de caminhos 44
4.2.2 Algoritmo com caminhos fixos 57
4.2.3 Algoritmo com caminhos por dia da semana 64
4.3 MELHORIAS LOCAIS 68
4.3.1 Algoritmo sem restrições de caminhos 69
4.3.2 Algoritmo com caminhos fixos 73
4.3.3 Algoritmo com caminhos por dia da semana 75
5 RESULTADOS **77**
5.1 DADOS GERADOS ALEATORIAMENTE 77
5.2 REDE REAL 81
5.3 DISCUSSÕES 101
6 CONCLUSÕES **108**
7 REFERÊNCIAS **112**

LISTA DE TABELAS

Tabela 1: Resumo dos principais trabalhos identificados 26

Tabela 2: Exemplo de demandas para serem roteirizadas na rede 53

Tabela 3: Relação de pernas na rede após alocação da demanda 1 54

Tabela 4: Relação de pernas na rede após alocação da demanda 2 55

Tabela 5: Relação de pernas na rede após alocação da demanda 3 57

Tabela 6: Exemplo de demandas para serem roteirizadas na rede para abordagem de caminhos fixos 62

Tabela 7: Relação de pernas anterior à alocação da demanda 3 para abordagem de caminhos fixos 62

Tabela 8: Relação de pernas após alocação da demanda 3 sem restrições 63

Tabela 9: Relação de pernas após alocação da demanda 3 com abordagem de caminhos fixos 64

Tabela 10: Relação de instâncias utilizadas 78

Tabela 11: Comparação de resultados entre heurística e modelos lineares 80

Tabela 12: Relação de filiais na rede real estudada 82

Tabela 13: Custos de transporte por tipo de linha 84

Tabela 14: Cenário de custos atual do caso real 85

Tabela 15: Resultados obtidos para os diferentes processamentos 85

Tabela 16: Quantidade de arcos utilizados para cada solução final encontrada 87

Tabela 17: Comparação de resultados entre trabalhos 104

Tabela 18: Comparação de tempos de processamento com demais trabalhos na literatura 106

LISTA DE FIGURAS

Figura 1: Exemplo de rede de terminais 12

Figura 2: Exemplo de consolidações de diversas demandas em uma rede de terminais 13

Figura 3: Exemplo de unicidade de caminhos. Todas as demandas com mesmo destino que chegam a um determinado terminal, devem seguir o mesmo caminho dali em diante. 15

Figura 4: Exemplo de pernas na rede 32

Figura 5: Relação entre variáveis temporais de demandas e pernas 34

Figura 6: Exemplo de criação de etiquetas de nós adjacentes 40

Figura 7: Exemplo de listagem de etiquetas 41

Figura 8: Exemplo de remoção de nós dominados 42

Figura 9: Pseudo-código para caminho mínimo com janelas de tempo 43

Figura 10: Pseudo-código para heurística construtiva 46

Figura 11: Cálculo de custo marginal 47

Figura 12: Pseudo-código para cálculo de custos de etiquetas 48

Figura 13: Verificação de factibilidade de consolidação 49

Figura 14: Pseudo-código para checagem de factibilidade temporal de consolidação de demandas 51

Figura 15: Pseudo código para carregamento de demanda na rede 52

Figura 16: Pequena rede de terminais 52

Figura 17: Rede após alocação da demanda 1 53

Figura 18: Rede após alocação da demanda 2 55

Figura 19: Rede após alocação da demanda 3 57

Figura 20: Exemplo de unicidade de caminhos para um destino *d* na rede 58

Figura 21: Exemplo de caminho pré-definido 59

Figura 22: Criação de *labels* para algoritmo com caminhos pré-definidos 59

Figura 23: Pseudo código para heurística construtiva com caminhos fixos 61

Figura 24: Pseudo código para carregamento de caminhos pré-definidos 61

Figura 25: Configuração da rede anterior à alocação da demanda 3 para abordagem de caminhos fixos 62

Figura 26: Configuração da rede após alocação da demanda 3 sem restrições 63

Figura 27: Configuração da rede após alocação da demanda 3 com abordagem de caminhos fixos 64

Figura 28: Criação de *labels* para caminhos por dia da semana 65

Figura 29: Pseudo código da heurística construtiva com caminhos por dia da semana 66

Figura 30: Pseudo código do carregamento de caminhos pré-definidos por dia da semana 66

Figura 31: Exemplo de consolidação com mudança no dia da semana 67

Figura 32: Exemplo de melhoria local 70

Figura 33: Exemplo de melhoria local 71

Figura 34: Exemplo de melhorias locais 72

Figura 35: Pseudo código para melhorias locais sem restrições de unicidade 73

Figura 36: Pseudo código para melhorias locais para caminhos fixos 75

Figura 37: Heurística de melhoria para caminhos por dia da semana..... 76

Figura 38: Redes das instâncias criadas 79

Figura 39: Filiais da rede real utilizada..... 83

Figura 40: Fluxos empregados na solução atual da empresa 84

Figura 41: Comparação entre solução atual e modelo sem restrições de unicidade de caminhos 87

Figura 42: Comparação entre solução atual e modelo com caminhos fixos 89

Figura 43: Comparação entre solução atual e modelo com caminhos por dia da semana..... 90

Figura 44: Resultados do cenário sem restrições de caminho 91

Figura 45: Variação frente ao mínimo encontrado para cenário sem restrições 92

Figura 46: Resultados do cenário com caminhos fixos 93

Figura 47: Variação frente ao mínimo encontrado para cenário com caminhos fixos 93

Figura 48: Resultados do cenário com caminhos por dia da semana 95

Figura 49: Variação frente ao mínimo encontrado para cenário com caminhos por dia da semana 95

Figura 50: Função objetivo com melhorias locais para caso sem restrições..... 96

Figura 51: Variação da função objetivo para melhorias locais no caso sem restrições 97

Figura 52: Função objetivo com melhorias locais para caso de caminhos fixos 98

Figura 53: Variação da função objetivo para melhorias locais no caso de caminhos fixos 99

Figura 54: Função objetivo com melhorias locais para caso de caminhos por dia da semana..... 100

Figura 55: Variação da função objetivo para melhorias locais no caso sem restrições 101

1 INTRODUÇÃO

No transporte rodoviário de cargas, existem dois tipos principais de contratação de frete: a carga lotação (*full truckload* ou FTL), caracterizada por uma origem e um único destino, em que a entrega completa a capacidade de um caminhão, e a carga fracionada (*less-than-truckload* ou LTL), em que cada remessa a ser entregue não é capaz, por si só, de completar um veículo.

No caso da lotação, geralmente é pago um valor fixo pela contratação de um veículo, independentemente da quantidade de mercadorias nele carregadas. Geralmente os prazos de entrega para esse tipo de transporte são menores, uma vez que o caminhão deverá seguir imediatamente do seu ponto de coleta para o destino.

Na carga fracionada, o valor pago é proporcional à quantidade de mercadorias embarcadas (geralmente medida em peso ou volume), e os prazos de entrega são maiores, uma vez que o operador encarregado do transporte deverá tentar consolidar o despacho com outras remessas com origens e destinos semelhantes.

Buscando uma melhor eficiência operacional, a transportadora de carga fracionada encontra um *trade-off* em relação a custo e nível de serviço ao realizar consolidações de carga. Uma alternativa seria, para manter um nível de serviço adequado, despachar imediatamente todas as entregas coletadas. Isso ocasionaria, porém, em veículos pouco carregados, aumentando os custos de operação como um todo.

Por outro lado, a transportadora poderia aguardar que outras remessas com mesma origem e mesmo destino surgissem e um caminhão completo fosse formado. Esse tipo de decisão operacional certamente diminuiria os custos do transporte, porém inevitavelmente iria afetar os prazos de entrega, principal medição do nível de serviço oferecido.

Uma solução encontrada por companhias que atuam nesse ramo é a criação de uma rede de terminais do tipo *hub-spoke*. Nesse tipo de rede, são caracterizados dois tipos de terminais:

- **Terminais de ponta** (*end of line* ou EOL): São terminais encarregados de coletas e entregas na região de sua proximidade. Dessa forma, as mercadorias que têm como origem ou destino uma região situada na sua zona de atuação serão coletadas ou entregues a partir desse terminal.
- **Terminais consolidadores** (*hub*): São terminais em que são realizadas operações de consolidação, ou seja, cargas chegam em determinados veículos, são descarregadas, reorganizadas e novamente despachadas rumo ao seu destino final. Terminais consolidadores também exercem o papel de terminais de ponta, uma vez que também deverão realizar entregas e coletas em clientes na sua zona de atuação.

Um exemplo de uma rede de terminais pode ser visto na Figura 1. Suponha que exista uma carga no ponto A com destino ao ponto D. O caminho percorrido por essa remessa será A->4->3->8->D. Primeiramente, será feita uma coleta a partir do terminal 4 na sua zona de atuação, passando também pelos pontos C e B, o que é geralmente feito em veículos menores.

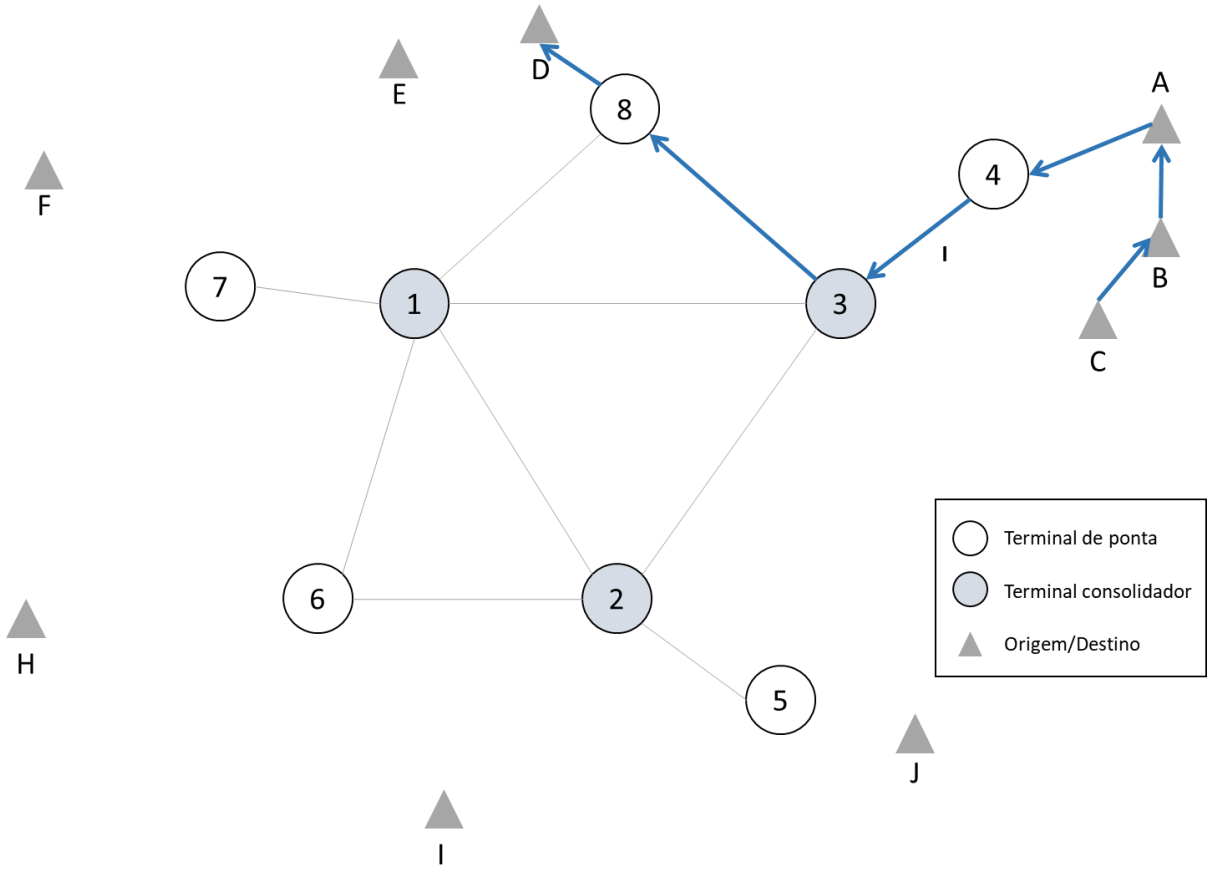

Figura 1: Exemplo de rede de terminais

Em seguida, a remessa será carregada em um veículo com destino ao terminal consolidador 3, junto com demais cargas coletadas na região do terminal de ponta 4, ainda que com destinos diferentes.

No terminal consolidador 3, é feita uma triagem e separação da carga de acordo com seu destino. A remessa tratada será enviada então ao terminal de ponta 8, cuja zona de atuação engloba o ponto D, destino final da remessa.

Na Figura 2 tem-se a mesma situação, mostrando diferentes demandas também presentes na rede. Os percursos de cada uma estão coloridos nos arcos da rede, enquanto os arcos são numerados com algarismos romanos. Dessa forma, é possível ver os benefícios em termos de eficiência que a rede de terminais pode trazer para a operação. No arco I, as demandas A-D e B-H são alocadas no mesmo veículo, mesmo tendo como destino terminais de ponta distintos, aumentando a sua ocupação em relação à situação em que elas seriam enviadas separadamente, levando a uma

redução de custos. O mesmo pode ser visto no arco II, em que as demandas B-H e J-F compartilham um mesmo veículo, ainda que tenham terminais de ponta de origem e destino distintos. Dado que as saídas de caminhões entre terminais são regulares nesse tipo de operação, é possível gerenciar e garantir os prazos de entrega, mantendo, portanto, um nível de serviço adequado.

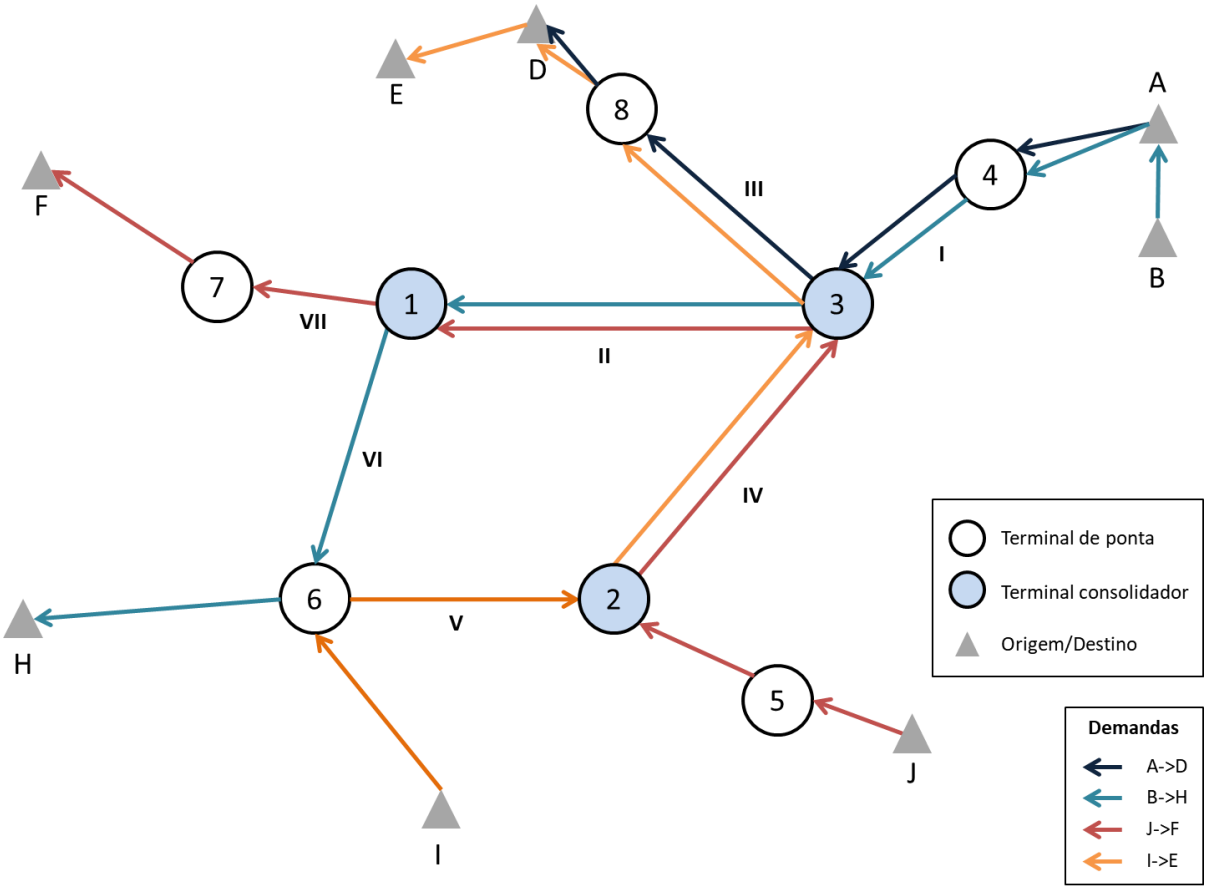

Figura 2: Exemplo de consolidações de diversas demandas em uma rede de terminais

Existem alguns outros aspectos em relação à rede de terminais desse gênero. Primeiramente, a operação de descarregamento, triagem e carregamento nos terminais em que há consolidação envolvem custos, denominados custos de movimentação. Embora sejam menores que os custos de transporte nos arcos da rede, esses ainda representam uma parcela significativa nos custos totais da operação. Além de custos, a passagem das encomendas pelos terminais também envolve tempos de movimentação, além de tempos de espera por caminhões, que devem ser considerados na avaliação de prazos de entrega.

Outra parcela significativa nos custos gerais do sistema é o referente a balanceamento de vazios. Terminais que tenham muitos veículos expedidos e poucos veículos chegando nas suas dependências geram uma demanda por veículos vazios, ao mesmo tempo que terminais que possuem mais veículos chegando do que saindo tenham um excesso de veículos vazios. Isso gera a necessidade de se equilibrar a rede, movimentando veículos vazios de terminais com excesso desses para terminais com esse tipo de demanda. Esse tipo de movimentação também gera uma parcela no custo total do sistema.

Embora no exemplo dado a remessa passe por somente um terminal consolidador, tão ou mais recorrentes são casos de consolidação em dois terminais consolidadores (caso da demanda J-F na Figura 2, por exemplo). Um maior número de consolidações deve aumentar os custos de movimentação, além de incorrer em maiores riscos de dano ou extravio de carga, de forma que são mais raros, o que faz com que diversos autores (como Jarrah *et al.*(2009) e Erera *et al.*(2012)) desconsiderem essa configuração nos seus trabalhos.

Além disso, sobre o formato da rede, são esperados arcos entre terminais de ponta e terminais consolidadores, além de ligações entre os diferentes terminais consolidadores da rede, porém são raros os casos de ligações entre terminais de ponta, uma vez que as demandas entre esses terminais dificilmente justificam a presença de um arco e de linhas regulares entre eles. Isso faz com que diversos autores, nos seus trabalhos, trabalhem com redes reduzidas, cortando, portanto, arcos considerados como pouco prováveis de serem utilizados (caso de Hoppe *et al*. (1999)). Além disso, existem autores que limitam as ligações de um terminal de ponta a um único terminal consolidador, simplificando a rede e reduzindo requisitos computacionais nas suas análises, caso de Lin *et al*. (2004).

Em relação à roteirização das remessas na rede, não é permitido que entregas que tenham mesma origem e mesmo destino sigam caminhos separados, de forma a impedir que haja um extravio, separação indevida de produtos ou chegada dessincronizada nos seus pontos finais.

Outro ponto semelhante nas redes de terminais de carga fracionada é a unicidade de caminhos ou árvores de caminhos. Como forma de simplificar as operações de triagem dentro dos terminais, evitando extravios de cargas, é estabelecido que todas as remessas que estejam em um terminal (seja ele consolidador ou de ponta) e que tenham um mesmo destino, independentemente da sua origem ou prazo de entrega, devem seguir o mesmo caminho dali em diante até o destino final. Isso faz com que os caminhos tenham um formato de árvore, conforme exibido na Figura 3. Existem alguns autores que discutem a existência dessa restrição, caso de Jarrah (2009), em que é apontado que, com o desenvolvimento de sistemas de informação e tecnologias de rastreamento mais avançadas, as empresas têm capacidade de flexibilizar essa limitação.

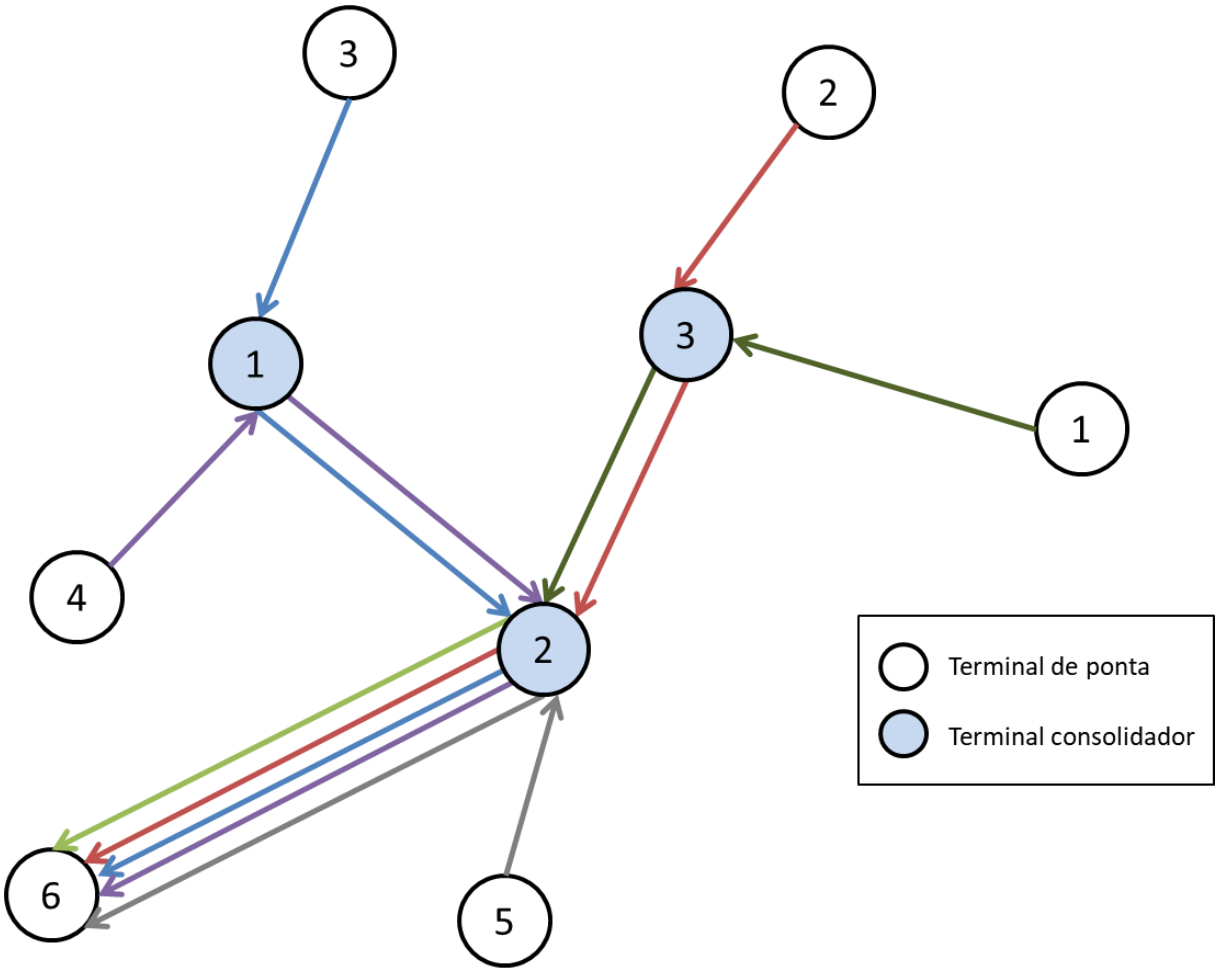

Figura 3: Exemplo de unicidade de caminhos. Todas as demandas com mesmo destino que chegam a um determinado terminal, devem seguir o mesmo caminho dali em diante.

Em relação ao estudo de redes de terminais para transporte de carga fracionada, podem ser delineadas três abordagens para o problema, conforme exposto em Crainic (2000):

- **Nível estratégico**: Trata da montagem da rede, como a localização física de cada um dos terminais, o seu porte, quais veículos serão utilizados, etc.
- **Nível tático**: No nível tático, é estudada a rede de serviços oferecida pelo operador, ou seja, quais serão as rotas dos veículos na rede e as suas regularidades, de forma a manter um nível de serviço adequado e uma redução de custos a partir de uma demanda esperada.
- **Nivel operacional**: No nível operacional, é abordado o dia-a-dia da operação, como atraso de veículos, o envio direto em caso de demanda acima do esperado e o gerenciamento e condução dos transportes e movimentação das cargas.

No presente trabalho, será dado enfoque ao nível tático, ou seja, a partir de uma rede estabelecida, deverão ser definidas quais as linhas de transporte que uma empresa deve oferecer de maneira a reduzir seus custos mantendo o nível de serviço desejado. Para tanto, primeiro é realizada uma revisão de trabalhos semelhantes, apresentada no Capítulo 2.

Em seguida, no Capítulo 3 são discutidas todas as premissas e delimitado o escopo do problema que deverá ser efetivamente tratado no presente trabalho, incluindo três diferentes abordagens para a restrição de unicidade de caminhos. Uma vez delimitado o escopo, são propostas três modelagens lineares mistas distintas para o problema, uma para cada abordagem.

Além das modelagens matemáticas exatas, é proposto um método heurístico para a resolução do problema, principal produto do presente trabalho, descrito no Capítulo 4. Em um primeiro momento, é realizada a descrição da fase inicial do método, que consiste em uma heurística construtiva, em que cada demanda do sistema é analisada e alocada na rede buscando a minimização de custos. De maneira similar ao capítulo anterior, são propostos três algoritmos distintos, um para cada abordagem para a restrição de unicidade de caminhos tratada.

Ainda no mesmo capítulo, é descrito um método de melhoria local em que, a partir dos resultados do algoritmo construtivo, são realizadas alterações nas soluções em busca de reduções de custo.

No Capítulo 5, são apresentados os resultados. Primeiramente, é realizada uma comparação entre os resultados obtidos para um dos modelos lineares mistos e o seu correspondente método heurístico. Dada a complexidade do problema e a dificuldade em aplicação do modelo linear misto a casos de grande porte, foram criadas 9 instâncias de pequeno porte, em que foi possível obter resultados e gerar comparações.

Em seguida, são apresentados os resultados da aplicação da heurística para dados de uma transportadora real, sendo avaliada a sua factibilidade em termos de aplicação a casos reais e em potenciais benefícios em termos de custo. Ao final do capítulo, é feita uma série de discussões a respeito dos resultados identificados.

Por fim, no Capítulo 6, são realizadas as considerações finais a respeito do trabalho, incluindo as suas principais conclusões e sugestões de continuidade da pesquisa no futuro.

2 REVISÃO BIBLIOGRÁFICA

Ao longo dos anos, diversos trabalhos trataram do problema de transporte de cargas fracionadas, considerando os mais diferentes aspectos e abordagens que esse problema pode comportar, desde a localização de terminais quanto à formação de uma rede de serviço de transporte, foco do atual trabalho.

Um dos primeiros trabalhos a tratar do planejamento de rotas em uma rede de terminais de carga fracionada no nível tático é o de Powell (1986), em que é utilizado o conceito de frequências mínimas. Nesse caso, os prazos de entrega são tratados de forma indireta, em que são estipulados volumes de veículos mínimos para determinados arcos da rede. Uma vez que esses volumes mínimos sejam atendidos, é assumido que os prazos de entrega não serão violados.

No trabalho, é apresentada uma modelagem matemática em programação linear que usa frequências mínimas, sendo também abordada a questão de transporte de vazios e equilíbrio de rede. Com base nesse modelo, é aplicada uma heurística a uma rede real de uma transportadora de cargas fracionadas de grande porte com 140 terminais.

O método proposto foi posteriormente revisitado em Powell *et al.* (1989), denominado também de projeto APOLLO, no qual foram aplicadas heurísticas de remoção e acréscimo de arcos a uma rede real de uma transportadora. O algoritmo proposto apresenta alternativas de redução de custos a um usuário, que toma a decisão final sobre as modificações na rede. Dessa forma, parte da complexidade do problema (como restrições legais, alfandegárias e comerciais), não incorporadas no modelo, são transferidas para um usuário externo.

A partir dos modelos de frequência mínima de Powell (1986) e Powell *et al.* (1989), Hellmuth (2004) propõe uma heurística de melhoria, em que são analisados os potenciais ganhos com a remoção ou inserção de arcos a partir de uma rede já montada. Seu trabalho foi aplicado a uma transportadora de cargas fracionadas brasileira com 18 terminais por todo o país.

Outro projeto que se baseia no modelo de frequências mínimas de Powell (1986) é Katayama *et al.* (2002), em que é aplicada uma relaxação lagrangeana em uma rede hipotética a fim de estudar efeitos nos requisitos computacionais.

Powell *et al.* (1983), por sua vez, embora ainda se utilizem do conceito de frequências mínimas, apresentam um modelo em que o prazo de entrega das remessas é calculado de maneira direta, ou seja, através da soma dos tempos de transporte em cada arco além dos tempos de parada em cada terminal. Para isso, é utilizada uma aproximação, em que o tempo de espera em um terminal é inversamente proporcional à frequência mínima do arco de saída do mesmo.

De maneira semelhante à Powell *et al.* (1989), é apresentada então uma heurística em que são analisados processos de criação/destruição de arcos, buscando encontrar economias nos custos totais do sistema. Para reduzir os requisitos computacionais, a soma de todos os tempos na rede foi considerada como uma parcela da função objetivo a ser reduzida, como forma de aproximar a restrição de máximo prazo de entrega.

Outro trabalho que trata os prazos de maneira direta é o de Hoppe *et al.* (1999), além de inserir frequências mínimas para determinados arcos na rede e balanceamento de vazios. Não são considerados, porém, tempos de espera por veículos nos terminais.

No trabalho, em especial, para garantir que os caminhos formem árvores, é realizada uma inversão, em que é fixado um nó na rede que seja destino de diversas demandas. A partir desse nó, são encontrados os caminhos mínimos considerando custos de transporte e manuseio até as suas diferentes origens. Invertendo as rotas obtidas, são obtidos caminhos que respeitam a unicidade de rotas, ou seja, em que demandas que chegam a um determinado nó e que possuem o mesmo destino, independentemente da origem, percorrerão o mesmo percurso dali em diante.

Buscando uma redução na demanda computacional, os autores também citam uma limpeza na rede, em que arcos que sejam considerados irrelevantes são excluídos de forma a melhorar o tempo de processamento. Como critério na decisão de exclusão,

são identificadas ligações entre terminais de ponta cuja demanda não ultrapassava um valor mínimo estabelecido.

Também é aplicada uma heurística de melhoria, em que são avaliados movimentos de exclusão ou inserção de novos arcos na rede, sendo feito iterativamente fixando também um nó de cada vez. Ao final, o modelo é utilizado em três instâncias de dados reais de transportadoras, com 48, 78 e 92 terminais.

Barcos *et al.* (2010) também consideram de forma direta os prazos de entrega em seu modelo baseado em algoritmos de colônia de formigas. Porém, assim como Hoppe *et al.* (1999), não consideram tempos de espera nos terminais para que um veículo seja despachado, o que pode limitar o seu modelo.

Em Barharnt *et al.* (2002), é abordado o tema de construção de uma rede de serviços para cargas expressas de modal aéreo, em que a roteirização é realizada em um horizonte de programação de um dia. Para a solução do problema, é feita uma separação entre a geração de rotas e a roteirização de remessas nessas rotas.

Para isso, são primeiramente determinadas quais são as rotas dos veículos sobre a rede, estabelecendo que deve haver um equilíbrio de vazios. Na definição das rotas, é utilizado um conjunto pré-estabelecido de percursos possíveis.

Uma vez que as rotas que deverão ser percorridas pelos veículos na rede já foram encontradas, as remessas do sistema deverão ser alocadas a alguma das rotas. No modelo, porém, diferentemente de modelos de transporte de carga fracionada, é permitido que produtos de mesma origem e mesmo destino tomem caminhos separados. Além disso, também não são previstas restrições temporais, uma vez que o horizonte de programação é de apenas um dia. O trabalho foi aplicado a uma rede de terminais de uma transportadora aérea de cargas, com um total de 258 nós.

Crainic *et al.* (1986) também fazem uma separação em dois subproblemas: a criação de uma rede e a posterior roteirização das demandas. No trabalho, é abordado o caso de uma ferrovia, semelhante ao observado no problema de cargas fracionadas na medida em que diversas demandas caracterizadas por uma origem e um destino deverão ser roteirizadas em uma rede de terminais. Uma vez criada a rede e

roteirizadas as demandas, é aplicado um algoritmo que estuda a mudança no nível de serviço de rotas na rede, avaliando então os impactos na função objetivo e, caso levem a uma redução, é iniciada uma nova iteração, criando uma nova rede de serviços e novas rotas para as demandas. Também é aplicado um algoritmo de geração de colunas buscando melhorar os resultados. Modelo semelhante é aplicado posteriormente em Crainic *et al.* (1988). Nesse trabalho, vale ressaltar, não é tratada a restrição de unicidade de caminhos a partir de um nó.

Outra aplicação a ferrovias é encontrada em Gorman (1998). Novamente, é feita uma separação entre a definição de quais serão as saídas de trens que formam a rede e roteirização das demandas a partir dessa decisão. Após a apresentação de uma modelagem em programação linear, é descrito um procedimento envolvendo algoritmo genético com busca tabu.

No algoritmo genético, as possibilidades de rotas percorridas pelos trens já são pré-definidas, que serão combinadas no algoritmo genético. Após a definição das saídas dos trens, é feita uma roteirização das demandas na rede, que se reduz a um problema de caminhos de custos mínimos. Estão previstas penalidades nos custos de uma dada solução, caso o prazo de entrega de uma determinada demanda não seja atingido, diminuindo a possibilidade de um indivíduo siga adiante na próxima geração do algoritmo genético nesse caso. Também é aplicada uma lógica de busca tabu, de forma a aumentar a variabilidade das soluções, sendo reportados melhores resultados.

Lin *et al.* (2008) apresentam um modelo matemático em programação linear em que também é realizada uma separação entre veículos e demandas, existindo, portanto, variáveis específicas para roteirizar as diferentes demandas na rede, bem como variáveis independentes das primeiras para roteirizar os veículos. Naturalmente, é presente uma relação entre as duas.

De forma semelhante, em Lin *et al.* (2004) também é aplicado um modelo em programação matemática para construção de uma rede de serviços de transporte. A fim de diminuir o espaço de busca, é imposto que cada terminal de ponta esteja ligado a somente um terminal consolidador, sendo gerados clusters de terminais.

Jarrah *et al.* (2009), como Lin *et al.* (2004) e Hoppe *et al.* (1999), também diminui o espaço de busca do seu algoritmo de forma a reduzir requisitos computacionais. Para isso, são delimitadas inicialmente diversas rotas factíveis para cada demanda. O processo de construção de rotas leva em consideração diversos aspectos:

- São permitidas somente rotas que passem por, no máximo, dois terminais consolidadores
- Rotas diretas entre dois terminais de ponta só são permitidas se existe um valor mínimo de demanda entre os dois
- Para um terminal consolidador ser selecionado, ele precisa estar mais próximo do destino do que a origem da carga
- A rota precisa respeitar o prazo de entrega estabelecido para o par O-D, considerando os tempos de consolidação nos terminais

No trabalho, são consideradas como factíveis as 100 rotas com menor distância que respeitam todas as restrições listadas acima. A partir desse espaço de soluções reduzido, é aplicado o modelo, em que o tratamento de prazos é feito através de uma rede temporal, na qual nós e arcos são definidos não só pela sua localização geográfica, mas também pela sua posição no tempo. Para tanto, é necessária uma discretização temporal, em que cada nó representa um dia no horizonte de programação.

Uma vez identificadas rotas factíveis, são geradas soluções iniciais sob a forma de árvores, ou seja, que garantem a restrição de unicidade de caminhos. Por fim, o modelo matemático se baseia em selecionar uma árvore de soluções para cada destino *s* na rede, de forma a minimizar custos de transporte, movimentação nos terminais e balanceamento de vazios. É então aplicada uma heurística do tipo *slope scalling* a uma rede com 725 nós (caracterizados por um terminal e um dia da semana), sendo ao final apresentados ganhos na operação original.

No trabalho de Erera *et al.* (2012), também é utilizado o conceito de rede temporal para abordar os prazos de entrega. Nesse caso, porém, foi utilizada uma discretização temporal maior, em que são considerados oito períodos ao longo do dia.

No trabalho, também é aplicada uma estratégia de redução do domínio de busca. Primeiramente, são encontradas diferentes rotas para uma determinada demanda, considerando caminhos mínimos de custo obtidos pelo algoritmo de Dijkstra. Nessa fase, não são consideradas restrições de tempo, de forma que posteriormente é necessário excluir todos os caminhos cujo tempo total violava o prazo total estabelecido para aquela demanda. São então geradas variações desses caminhos mínimos considerando tempos de espera em terminais consolidadores.

O modelo matemático proposto se baseia na seleção de um caminho para cada demanda dentro do seu domínio de caminhos potenciais, considerando custos de transporte, consolidação e equilíbrio de equipamentos (vazios). Dado o tamanho do problema tratado, é discutido um algoritmo de reotimização, em que, a partir de uma dada solução inicial, são procuradas melhorias locais analisando uma demanda por vez, mantendo todas as outras fixas em suas rotas.

Em relação à unicidade de caminhos, é realizada uma avaliação sobre a flexibilização desse tipo de restrição, tanto para variação de redes ao longo de uma semana quanto para o não cumprimento da restrição.

Ao final, é feita uma aplicação a instâncias de uma rede real de uma transportadora de carga fracionada, contendo mais de 160 terminais, dos quais 60 consolidadores. Analisando as variações em relação a restrição de unicidade de caminhos, foi apontado que a utilização de caminhos variando ao longo da semana produzem resultados mais promissores do que o simples não cumprimento da restrição.

Em Zhang *etal.* (2010), são abordados diversos aspectos diferentes em relação ao planejamento de redes de transporte de carga fracionada. Primeiramente, é previsto um método de cálculo de custos a partir de uma rede estabelecida de saídas de caminhões, que inclui o balanceamento de vazios para equilibrar saídas e chegadas de caminhões em todos os terminais.

Além disso, é proposta um planejamento de rede de transporte de carga fracionada dinâmico, que seria processado com uma regularidade maior do que o proposto em outros trabalhos, baseado na situação atual da rede em um dado instante de tempo.

Para isso, são apresentados dois métodos: uma busca local realizada através de programação linear inteira mista e uma heurística gulosa de melhoria.

Nos dois métodos, são utilizadas redes temporais (semelhante ao visto nos trabalhos de Jarrah *etal.* (2009) e Erera *etal.* (2012)), sendo o tempo discretizado de hora em hora ao longo de cada dia. Tanto para a modelagem em programação inteira quanto para a heurística, parte-se de uma solução inicial, sendo então realizadas buscas locais analisando um certo conjunto de demandas, fixando todas as demais nas suas posições na rede, de forma similar a Hoppe *etal.* (1999) e Erera *etal.* (2012).

No caso da modelagem inteira, é previsto um conjunto fechado de rotas possíveis para cada demanda, reduzindo o espaço de soluções e aumentando a eficiência computacional, operação semelhante a Hoppe *etal.* (1999), Gorman (1998), Jarrah *etal.* (2009) e Erera *etal.* (2012).

Já para a heurística, são procurados caminhos mínimos de custos marginais para cada demanda, considerando o custo adicional de se inserir a demanda analisada em um veículo que já percorre um dado arco. Nesse cálculo não é inserido, porém, o custo de movimentação nos terminais, sendo considerado somente o custo de transporte nos arcos.

O processo de seleção de qual será a demanda analisada é randomizada, sendo ponderado primeiramente pelo tempo restante até o prazo e, em um segundo nível, pelo seu volume. Esse procedimento de escolha deverá se repetir até que não sejam mais encontradas melhorias no sistema, ou que um dado critério de parada indique seu fim.

Como em Jarrah *etal.* (2009), também é discutida a restrição de caminhos percorridos em forma de árvore na rede. É mencionado no trabalho que, com o advento de novas tecnologias de etiquetagem e leitura, é possível para os operadores separarem no terminal em rotas diferentes demandas que possuam origens diferentes, ainda que tenham o mesmo destino. Tanto para a modelagem em programação inteira quanto para a heurística, foram criadas duas variações: uma que atende a necessidade de caminhos em forma de árvore e outra que relaxa essa restrição.

Ao final, foram aplicados os dois métodos a uma rede de terminais real de uma transportadora com 161 nós e aproximadamente 20.000 remessas. Foram encontradas reduções de custos entre 7% e 11% e, como esperado, os modelos em que a restrição de caminho em forma de árvore foi relaxada apresentou maiores ganhos.

Também no trabalho, é dedicada uma Seção a respeito de modelagem de planejamento de rede de transportes de carga fracionada sob o aspecto estocástico, e não determinístico, como é encontrado nas demais referências. Dada a maior dificuldade na resolução do problema estocástico, é tratada somente uma parte da rede original da transportadora, apresentando uma menor granularidade em respeito ao tempo, além de apresentar outras limitações em relação ao modelo determinístico proposto.

Outro algoritmo baseado no conceito de custos marginais foi utilizado no trabalho de Vasco *et al.* (2016), em que é tratado o problema de alocação de veículos a uma rede. Embora a aplicação tenha sido feita para um sistema de frete lotação, o algoritmo proposto também faz uma ordenação de tarefas (demandas), inserindo uma a uma e minimizando custos marginais. Na ordenação de demandas, é utilizado um valor aleatório, buscando gerar diferentes casos, inspirada na meta-heurística GRASP.

Em Erera *et al.* (2013), um algoritmo de custos marginais também é aplicado, embora para um aspecto diferente do planejamento de redes de carga fracionada, em que é analisada a programação de motoristas para uma rede pré-estabelecida.

Além de Hellmuth (2004), outra aplicação no Brasil foi feita por Silva (2010), que trata no seu trabalho, principalmente, do aspecto estratégico do transporte de cargas fracionadas, ou seja, no posicionamento e criação de uma rede de terminais. É inclusa, porém, uma Seção destinada a planejamento e roteirização das diferentes demandas que o sistema deve tratar na rede. Para isso, as demandas são ordenadas em ordem decrescente e o caminho mínimo de custos é encontrado. Arcos que já tenham sido utilizados apresentam custos zerados nas iterações futuras, de forma a incentivar a consolidação das cargas. Não é explicitada uma forma de tratamento para prazos de entrega, porém.

Prado (2017) também trata através de uma visão estratégica as operações de carga fracionada. No trabalho, é proposta uma modelagem que processa simultaneamente a localização de terminais na rede e a roteirização de demandas entre os mesmos. No modelo, porém, é tratado somente um período, que impede a avaliação de consolidação de cargas através da espera nos terminais. Dessa forma, cada carga deve ser despachada imediatamente após o seu recolhimento. Também são estabelecidos formatos para as rotas, que faz com que consolidações por mais de 3 terminais sejam difíceis de acontecer. É através também desse filtro de rotas factíveis que são tratados os prazos de entrega, impedindo que rotas muito longas sejam montadas e os prazos possam ser feridos. O trabalho foi aplicado a uma transportadora real no Estado de São Paulo, com um total de 15 terminais considerados.

Ao final, é possível observar na Tabela 1 um resumo dos principais trabalhos identificados, considerando qual a tratativa para os prazos de entrega, a restrição para unicidade de caminhos, o equilíbrio de veículos vazios na rede, a utilização ou não de buscas locais e uma eventual estratégia para reduzir a dificuldade do problema.

Tabela 1: Resumo dos principais trabalhos identificados

Trabalho	Prazos	Unicidade de caminhos	Vazios	Busca Local	Redução dificuldade
Powell (1986)	Indiretos - Frequências mínimas	Considerada	Considerado	Adiciona/ Cancela arcos	-
Powell *et al.* (1983)	Diretos com uso de frequências mínimas	Considerada	Considerado	Adiciona/ Cancela arcos	-
Powell *et al.* (1989)	Indiretos - Frequências mínimas	Considerado	Considerado	Adiciona/ Cancela arcos	-
Hoppe *et al.* (1999)	Diretos com uso de frequências mínimas	Considerada	Considerado	Fixa rede e analisa demanda a demanda	Rede com arcos removidos
Barcos (2010)	Tempos diretos	Não considerada	Considerado	-	Rede com arcos removidos

Trabalho	Prazos	Unicidade de caminhos	Vazios	Busca Local	Redução dificuldade
Hellmuth (2004)	Indiretos - Frequências mínimas	Considerada	Considerado	Adiciona/ Cancela arcos	-
Crainic *et al.* (1986) Crainic *et al.* (1988)	Tempos diretos	Não considerada	Considerado	-	-
Gorman (1998)	Tempos diretos	Não considerada	Não considerado	-	Rotas pré-definidas
Lin *et al.* (2004)	Tempos diretos	Não considerada	Não considerado	-	Rede com arcos removidos
Lin *et al.* (2008)	Tempos diretos	Considerada	Considerado	-	-
Jarrah *et al.* (2009)	Redes temporais	Considerada e flexibilizada	Considerado	Fixa rede e analisa demanda a demanda	Rotas pré-definidas
Barharnt *et al* (2002)	-	-	Considerado	-	-
Silva (2010)	-	Não considerada	Não considerado	Adiciona/ Cancela arcos	Rotas pré-definidas
Prado (2017)	Tempos diretos	Não considerada	Considerado	-	Rotas pré-definidas
Erera *et al.* (2012)	Redes temporais	Considerada	Considerado	Fixa rede e analisa demanda a demanda	Rotas pré-definidas
Zhang *etal.* (2010)	Redes temporais	Considerada e flexibilizada	Considerado	Fixa rede e analisa demanda a demanda	Rotas pré-definidas

3 DESCRIÇÃO DO PROBLEMA E MODELAGEM MATEMÁTICA

No presente trabalho, será abordado o nível tático do problema de transporte de cargas fracionadas, ou seja, a construção de rotas de veículos e remessas na rede, minimizando os custos operacionais totais e garantindo atendimento ao nível de serviço desejado. O problema pode ser classificado como pertencente à classe de *Service Network Design*, descrita por Crainic (2000). Também é possível aproximá-lo ao *Multicommodity Flow Problem* (Ahuja et al. (1993)), em que diversas demandas com diferentes nós de origem e destino devem ser roteirizadas em um grafo com custos definidos.

Para isso, será considerado como dada a rede de terminais de uma transportadora de cargas fracionadas, incluindo a localização de cada terminal, bem como os custos e tempos associados a transporte entre os diferentes pontos da rede e a movimentação de mercadorias dentro dos terminais.

Também será considerada como entrada do problema a relação de todas as demandas de transporte, caracterizadas por um terminal de origem, um terminal de destino e um tamanho. Para cada par origem-destino, também estará estipulado o nível de serviço desejado, caracterizado pelo prazo máximo de entrega desde a entrada no sistema.

A partir de todos esses dados, pretende-se construir uma heurística construtiva que permita identificar quais são as rotas percorridas por veículos na rede e as suas frequências, bem como qual caminho cada demanda deverá percorrer desde a sua origem até o destino, buscando melhorias do ponto de vista de custo e respeitando as restrições de prazos de entrega. Também será proposta uma heurística de melhoria, visando obter reduções nos custos a partir de uma primeira solução estabelecida.

Um dos aspectos principais da heurística proposta é que ela não se usará de uma rede *temporal* para roteirização das demandas, em que cada nó é caracterizado não só pela sua posição física, mas também por uma posição temporal. Esse tipo de abordagem, utilizada nos trabalhos de Jarrah *et al.* (2009), Erera *et al.* (2012) e Zhang (2010), requer necessariamente uma discretização do tempo, cuja granularidade pode

ser objeto de discussão, uma vez que afeta o resultado final e também o tempo de processamento do modelo. O algoritmo do presente trabalho tratará o tempo de maneira contínua, através de um método de caminhos mínimos com janelas de tempo, eliminando a necessidade de uma discretização temporal.

Em relação aos tipos de veículos, deverá ser considerada uma frota homogênea, com capacidade fixa e igual para todos os veículos, de maneira semelhante aos trabalhos de Powell (1986), Jarrah *et al.* (2009), Erera *et al.* (2012), Hellmuth (2004), Zhang (2010) e Hoppe *et al.* (1999). Essa simplificação faz com que seja necessária uma adaptação nos dados de entrada, dado que operações reais contam com uma grande lista de veículos utilizados.

Também não serão feitas restrições relacionadas aos tipos de terminais, se consolidadores ou de ponta, como feito nos trabalhos de Jarrah *et al.* (2009) e Lin (2008). Dessa forma, não só todas as conexões entre terminais são passíveis de serem utilizadas, como não é feita uma distinção entre terminais de ponta e consolidadores. Evita-se, portanto, a necessidade de classificar terminais como consolidadores ou de ponta, o que poderia gerar uma perda no potencial de redução de custos do sistema. O algoritmo tem, portanto, liberdade de fazer consolidações em qualquer terminal, embora alguns terminais exibam uma “vocação” para realizar consolidações, atrelada aos seus custos unitários de consolidação, posição geográfica e volume de demanda alocado.

Da mesma forma que não serão classificados terminais como consolidadores ou de ponta, não serão feitas limitações a quais tipos de arcos poderão ser utilizados pelas demandas, como também observado nos trabalhos de Jarrah *et al.* (2009) e Lin (2004). O algoritmo deverá ter liberdade de escolher quaisquer arcos, garantindo que uma melhor solução não será eliminada antes mesmo do processamento do modelo. Naturalmente, terminais com pouco volume de carga sendo destinado ou originado de si dificilmente terão muitos arcos entre si, dado que provavelmente o seu custo seria muito elevado.

Todos os terminais também terão capacidade infinita de processamento, ou seja, não haverá limitação na capacidade de processamento de cada terminal, bem como

qualquer restrição ligada a horário ou dias de funcionamento. Em operações reais, os terminais muitas vezes não trabalham 24 horas por dia e têm limitações de capacidade. O resultado do modelo sem as restrições pode ser um indicativo, porém, de qual seria a capacidade ideal de operação e os horários de funcionamento de maneira a reduzir seus custos na rede. Esse pode ser outro ponto de extensão do trabalho no futuro, uma vez que os tempos de estadia de cada remessa podem variar de acordo com o seu horário de chegada no terminal e o seu decorrente horário de funcionamento e capacidade, caso essas restrições sejam consideradas.

Também não será tratado no presente trabalho o rebalanceamento de veículos vazios na cadeia. No trabalho de Hellmuth (2004), é mencionado que a parcela decorrente desse tipo de movimentação corresponde a menos de 10% dos custos totais operacionais, e foi entendido que a sua resolução poderia ser feita à parte em um modelo de programação linear. Esse poderá ser um ponto interessante de extensão para futuros trabalhos, porém.

Além disso, o presente trabalho deverá considerar diferentes variações para a restrição da unicidade de caminhos:

1. **Algoritmo sem restrições de caminhos:** Nesse modelo, não há necessidade de demandas que estejam em um determinado terminal e compartilhem o mesmo destino sigam o mesmo caminho dali em diante. Dessa maneira, cada uma pode seguir sua própria rota, fazendo com seja a variante mais flexível do modelo.
2. **Algoritmo com caminhos fixos para par origem-destino:** Nessa variante, a restrição de unicidade de caminhos é totalmente respeitada, ou seja, todas as demandas que estejam em um determinado terminal nas suas rotas e tenham mesmo destino deverão obrigatoriamente seguir a mesma rota.
3. **Algoritmo com caminhos por dia da semana para pares origem-destino**: Será proposto um modelo mais flexível em relação à segunda variante, em que a estrutura de árvore nos caminhos ainda permanece, mas que pode ser variada de acordo com o dia da semana. Esse tipo de abordagem como uma

alternativa ao caso sem restrições é discutido em detalhe no trabalho de Jarrah *et al.* (2009).

Para cada uma das variações mencionadas, deverá haver uma correspondente versão de heurística construtiva (Seção 4.2) e de heurística de melhoria (Seção 4.3). Os modelos matemáticos propostos para cada uma das variações também podem ser observados nas Seções 3.1,3.2 e 3.3.

3.1 Modelo sem restrições de caminhos

Nessa Seção é apresentada uma proposta de modelagem matemática em programação linear para a roteirização de demandas na rede de terminais considerando que não há restrições de unicidade de caminhos, ou seja, cada demanda pode seguir livremente qualquer percurso na rede, independentemente de quais outras rotas as demais demandas com o mesmo destino seguem no sistema.

Além de auxiliar no entendimento do problema, o modelo proposto será utilizado para realizar comparações de resultados entre o método heurístico e a solução ótima (Seção 5.1). Dada a complexidade do problema, isso só será possível para instâncias de pequeno porte, para as quais é possível chegar a uma solução ótima através do modelo linear.

A modelagem proposta se baseia, assim como a heurística posteriormente apresentada, no conceito de **perna**, caracterizada por um veículo que atravessa um arco da rede. Cada perna deverá ter um horário de saída do nó de origem e um horário de chegada no nó de destino, relacionados a partir do tempo de transporte do arco.

Na Figura 4, por exemplo, é mostrada uma pequena rede com 4 nós. Nela, é possível distinguir três diferentes pernas de transporte: uma entre o arco *(i,j)*, outra no arco *(j,k)* e finalmente uma no arco *(j,l)*. Cada perna tem um instante de saída independente das demais pernas, e é capaz de consolidar um máximo de demandas correspondente ao valor nominal da capacidade dos veículos. Nada impediria, na operacionalização da solução, de um mesmo veículo executar as pernas de transporte (*i,j*) e (*j,k*) se fosse temporalmente factível. Ainda assim, o problema estaria tratando de duas pernas de transporte distintas, mesmo que executadas pelo mesmo veículo.

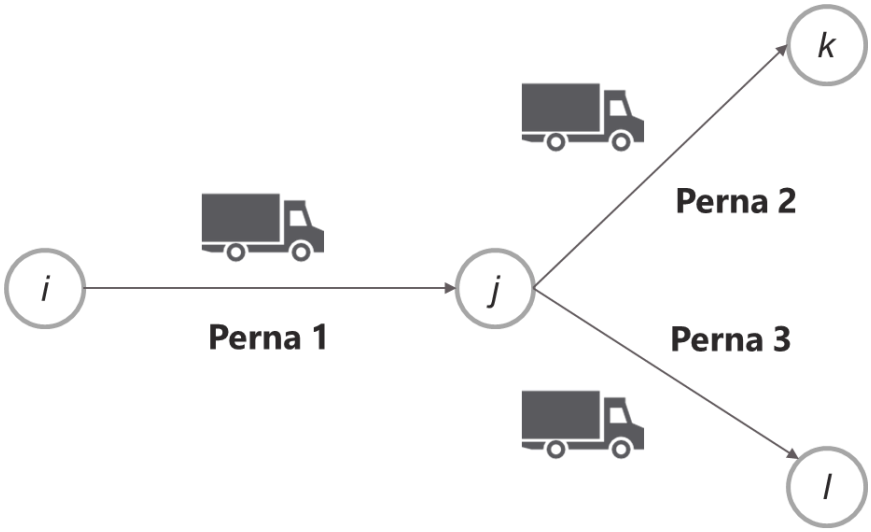

Figura 4: Exemplo de pernas na rede

Também é feita uma separação entre as rotas seguidas pelas demandas na rede e os veículos que atravessam os terminais, de maneira semelhante ao observado no trabalho de Lin *et al.* (2008). Existe, naturalmente, uma conexão entre as duas, de forma a garantir que uma demanda está sempre associada a um veículo (na verdade, a uma perna) ao percorrer algum arco da rede.

Os índices utilizados para o modelo são os apresentados abaixo. Em especial, é importante notar que o número de pernas na rede que poderão ser criadas é estabelecido *a priori*. A partir da experiência do analista, é possível estipular um número grande o suficiente de forma a não afetar a solução ótima do problema, recordando porém que um aumento no número de pernas resulta em uma maior dificuldade no processamento. Um indicativo de que não houve influência na seleção do número de pernas frente ao ótimo é a presença de pernas ociosas em todos os arcos, mostrando que não houve limitação de recursos no processo de busca de soluções.

<u>Índices e parâmetros</u>

q: Demandas

i,j: Terminais na rede, que são entendidos também como nós do grafo

l: Pernas

Os parâmetros de entrada para o modelo são os seguintes:

e_q	$instante\ de\ entrada\ da\ demanda\ q\ na\ rede$
p_q	$instante\ máximo\ de\ entrega\ da\ demanda\ q\ na\ rede$
s_q	$tamanho\ da\ demanda\ q$
f_i^q	$parâmetro \begin{cases} -1, se\ i\ é\ origem\ da\ demanda\ q \\ 1, se\ i\ é\ destino\ da\ demanda\ q \\ 0, para\ os\ demais\ terminais \end{cases}$
β_{ij}^l	$parâmetro\ binário \begin{cases} 1, se\ a\ perna\ l\ atravessa\ o\ arco\ (i,j) \\ 0, caso\ contrário \end{cases}$
c_{ij}	$custo\ de\ um\ veículo\ atravessar\ o\ arco\ (i,j)$
t_{ij}	$tempo\ de\ um\ veículo\ atravessar\ o\ arco\ (i,j)$
m_i	$custo\ de\ movimentação\ no\ terminal\ i$
h_i	$tempo\ de\ movimentação\ no\ terminal\ i$
or_q	$origem\ da\ demanda\ q$
ds_q	$destino\ da\ demanda\ q$
Cap	$capacidade\ do\ veículo$
N	$parâmetro\ suficientemente\ grande\ para\ linearizar\ equações,$ $podendo\ ser\ utilizado\ o\ número\ total\ de\ demandas$

Como mencionado anteriormente, é estabelecido um número de pernas *a priori* no sistema. Dessa maneira, é possível construir o parâmetro β_{ij}^l, que indica a qual arco da rede uma determinada perna *l* percorre.

<u>Variáveis de decisão</u>

As variáveis de decisão do modelo são as expostas abaixo. Em termos gerais, o modelo incorpora as seguintes decisões, baseadas nas variáveis:

1. Quais pernas deverão ser ativadas (z_l)
2. Para cada perna, qual horário de saída do nó de origem (e consequentemente, de chegada no nó de destino) (d_l)
3. Quais demandas deverão estar associadas a cada arco (x_l^q)

$$x_l^q: \quad variável\ binária \begin{cases} 1, se\ a\ demanda\ q\ usa\ a\ perna\ l \\ 0, caso\ contrário \end{cases}$$

$$a_i^q: \quad instante\ de\ chegada\ da\ demanda\ q\ no\ terminal\ i$$

$$d_i^q: \quad instante\ de\ saída\ da\ demanda\ q\ no\ terminal\ i$$

$$a_l: \quad instante\ de\ chegada\ da\ perna\ l$$

$$d_l: \quad instante\ de\ saída\ da\ perna\ l$$

$$z_l: \quad variável\ binária \begin{cases} 1, se\ a\ perna\ l\ é\ utilizada \\ 0, caso\ contrário \end{cases}$$

Percebe-se que existem dois tipos de variáveis associadas a instantes de tempo de saída e chegada em nós da rede: um para demandas (a_i^q e d_i^q) e outro para pernas (a_l e d_l).

Naturalmente, deve existir uma correlação entre instantes de tempo de pernas e demandas. Todas as demandas associadas a uma perna deverão necessariamente ter os seus instantes de saída do nó de origem e de chegada no nó de destino equivalentes aos da perna.

Na Figura 1Figura 5 pode ser observado um exemplo. Uma demanda *q* atravessa dois arcos (*i,j)* e (*j,k*) na rede, estando associada a duas pernas, L_1 e L_2, respectivamente.

No esquema apresentado, o instante de tempo de saída dos nós *i* e *j* da demanda *q* (d_i^q e d_j^q) deverão ser iguais aos instantes de saída das pernas (d_{L_1} e d_{L_2}) para cada arco. O mesmo é válido para os instantes de chegada nos nós *j* e *k* (a_j^q e a_k^q), que deverão ter os mesmos valores das variáveis das pernas (a_{L_1} e a_{L_2}).

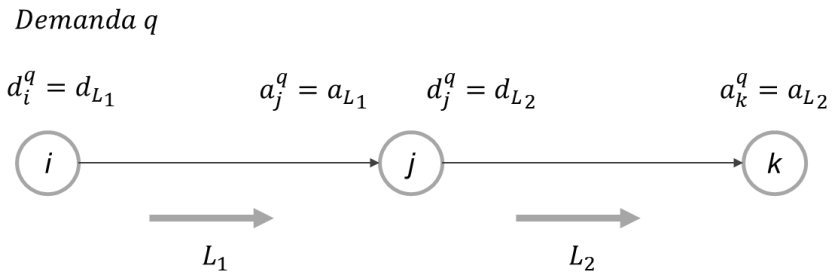

Figura 5: Relação entre variáveis temporais de demandas e pernas

A função objetivo, por sua vez, é construída da seguinte forma:

$$min \sum_{i} \sum_{j} \sum_{l} z_l \beta_{ij}^{l} \, c_{ij} + \sum_{q} \sum_{i} \sum_{j} \sum_{l} x_l^q \beta_{ij}^{l} m_j s_q$$

A primeira parcela está relacionada aos custos de transporte de veículos na rede. Já a segunda diz respeito aos custos de movimentação nos terminais.

Por fim, as restrições que o problema deve respeitar são descritas a seguir:

$$\sum_{q} x_l^q s_q \leq Cap \qquad \forall l \qquad (1)$$

$$N z_l \geq \sum_{q} x_l^q \qquad \forall l \qquad (2)$$

$$\sum_{l} \sum_{i} x_l^q \beta_{ij}^{l} - \sum_{l} \sum_{k} x_l^q \beta_{jk}^{l} = f_j^q \qquad \forall j, q \qquad (3)$$

$$\sum_{l} \sum_{j} x_l^q \beta_{ij}^{l} \leq 1 \qquad \forall i, q \qquad (4)$$

$$d_i^q - a_i^q \geq h_i \sum_{l} \sum_{j} x_l^q \, \beta_{ji}^{l} \qquad \begin{cases} \forall q, i \\ i \neq or_q \\ i \neq ds_q \end{cases} \qquad (5)$$

$$a_i^q \leq p_q \qquad \forall q, i = ds_q \qquad (6)$$

$$a_i^q \geq e_q \qquad \forall q, i = or_q \qquad (7)$$

$$d_l + \sum_{i} \sum_{j} \beta_{ij}^{l} t_{ij} = a_l \qquad \forall l \qquad (8)$$

$$a_i^q = \sum_{l} \sum_{j} x_l^q \beta_{ji}^{l} a_l \qquad \forall i, q \qquad (9)$$

$$d_i^q = \sum_{l} \sum_{j} x_l^q \beta_{ij}^{l} d_l \qquad \forall i, q \qquad (10)$$

$$x_l^q, z_l \text{ binárias} \qquad (11)$$

$$a_i^q, d_i^q, a_l, d_l \geq 0 \qquad (12)$$

As restrições (1) garantem que a soma dos volumes de demandas associados a uma perna não supera a sua capacidade. Já as restrições (2) determinam que uma perna

é utilizada caso tenha alguma demanda associada a ela. A equação (3) está associada à continuidade dos nós que uma demanda percorre.

As restrições (4) impõem que a demanda só pode ser atribuída a no máximo uma única perna em um determinado arco. Já as restrições (5) garantem os tempos de movimentação e a continuidade dos tempos nas rotas. As restrições (5), (6) e (7) garantem os tempos de movimentação, prazos de entrega e instantes de entrada no sistema para as demandas. A equação (8) relaciona os instantes de chegada e saída da perna, respeitando os tempos de transporte nos arcos. Por fim, as equações (9) e (10) associam os instantes de chegada e saída de uma demanda às pernas às quais ela está associada. Dado que elas não são lineares, é necessário criar as equações (9a), (9b), (10a) e (10b), que linearizam a relação entre variáveis, conforme apresentado abaixo.

$$a_i^q \leq \sum_l \sum_j \beta_{ij}^l (1 - x_l^q) N + a_l \qquad \forall i, q \qquad \text{(9a)}$$

$$a_i^q \geq \sum_l \sum_j -\beta_{ij}^l (1 - x_l^q) N + a_l \qquad \forall i, q \qquad \text{(9b)}$$

$$d_i^q \leq \sum_l \sum_j \beta_{ij}^l (1 - x_l^q) N + d_l \qquad \forall i, q \qquad \text{(10a)}$$

$$d_i^q \geq \sum_l \sum_j -\beta_{ij}^l (1 - x_l^q) N + d_l \qquad \forall i, q \qquad \text{(10b)}$$

3.2 Modelo com caminhos fixos

A segunda variação de modelo proposta deverá comportar a restrição de que, a partir de um dado nó, todas as demandas que compartilham o mesmo destino deverão percorrer o mesmo caminho daquele ponto em diante.

Para tanto, é necessário estabelecer que haverá um agrupamento de demandas de acordo com o seu destino final. Chamemos de S_d o conjunto de todas as demandas que tenham como destino final *d.* Nesse caso, é estabelecida uma variável binária que diz respeito ao caminho que as demandas deverão percorrer.

$$z_{ij}^{d}: \quad variável\ binária \begin{cases} 1, se\ as\ demandas\ com\ destino\ d\ seguem\ para\ j\ a\ partir\ de\ i \\ 0, caso\ contrário \end{cases}$$

A partir dessa variável, são inseridas duas novas restrições no modelo, além das restrições (1) a (12) apresentadas na Seção 3.1.

$$x_l^q \beta_{ij}^l \leq z_{ij}^d \qquad \forall i, j, l \quad \forall q \in S_d \qquad (13)$$

$$\sum_j z_{ij}^d \leq 1 \qquad \forall i, d \qquad (14)$$

As restrições 13 impõem que cada demanda só pode seguir, a partir de um nó *i,* para um nó *j* caso o próximo nó no caminho determinado para todas as demandas do seu conjunto S_d seja esse. Já as restrições (14) colocam que somente pode haver um único nó *j* válido para um destino *d* e um determinado nó *i*.

3.3 Modelo com caminhos por dia da semana

De maneira análoga à variação do problema em que todas as demandas que compartilham destino e um nó intermediário deverão seguir o mesmo caminho, também é proposta uma variante do modelo que permite uma flexibilização por dia da semana. Dessa maneira, somente demandas que cheguem no mesmo dia da semana em um determinado nó e que tenham o mesmo destino deverão seguir o mesmo percurso dali em diante.

Nesse caso, é necessário estabelecer um novo índice que será utilizado no sistema:

<u>Índice</u>

k: dias da semana no horizonte de programação tratado

Também são necessárias duas novas variáveis de decisão:

$$z_{ij}^{kd}: \quad variável\ binária \begin{cases} 1, se\ as\ demandas\ com\ destino\ d\ que\ chegam\ em\ i\ no\ dia\ da \\ semana\ k\ seguem\ por\ j \\ 0, caso\ contrário \end{cases}$$

$$w_{ik}^{q}: \quad variável\ binária \begin{cases} 1, se\ a\ demanda\ q\ chega\ no\ nó\ i\ no\ dia\ da\ semana\ k \\ 0, caso\ contrário \end{cases}$$

Em termos de restrições, as restrições de (1) a (12) descritas na Seção 3.1 deverão se manter, acrescentando as seguintes:

$$x_l^q \beta_{ij}^l \leq z_{ij}^{kd} + N(1 - w_{ik}^q) \qquad \forall i, j, l, k \quad \forall q \in S_d \qquad (15)$$

$$\sum_j z_{ij}^{kd} \leq 1 \qquad \forall i, d, k \qquad (16)$$

$$\sum_k w_{ik}^q \leq 1 \qquad \forall i, q \qquad (17)$$

$$\sum_k k.w_{ik}^q \geq a_i^q \qquad \forall i, q \qquad (18)$$

$$\sum_k (k-1).w_{ik}^q \leq a_i^q \qquad \forall i, q \qquad (19)$$

As restrições (15) e (16), de maneira semelhante ao modelo anterior, que uma demanda a partir de um nó *i* que tenha um destino *d* só pode seguir por um único nó *j.* Com a adição da parcela $N(1 - w_{ik}^q)$, tem-se que essa obrigatoriedade só é válida caso a demanda chegue no nó *i* no dia de semana k.

Já as restrições (17), (18) e (19) permitem que a variável w_{ik}^q seja definida. A primeira determina que uma demanda só pode chegar em um único dia da semana para cada nó.

Já as equações as restrições (18) e (19) fazem a relação entre o dia da semana *k* e a variável contínua que define a chegada da demanda no nó *i*, através de uma discretização. A primeira tenta identificar um *k* que seja o inteiro arredondado para cima em relação ao instante de chegada a_i^q. Já a segunda identifica o (*k*-1) inteiro arredondado para baixo de a_i^q. Só haverá um *k* que satisfaça as duas equações ao mesmo tempo, ao qual a variável w_{ik}^q terá valor 1, sendo 0 para os demais *k*. É importante ressaltar que essas equações são válidas para instantes de tempo medidos em dias. Caso seja medido em horas, por exemplo, o termo a_i^q deveria ser dividido por 24.

4 MÉTODO DE SOLUÇÃO

No presente trabalho, para a resolução do problema de transporte de carga fracionada em uma rede de terminais, além do modelo em programação linear inteira mista proposto no Capítulo 3, foi desenvolvido um algoritmo que se utiliza do conceito de **custos marginais** por demanda, semelhante ao proposto em Zhang (2010).

No método proposto, cada demanda (caracterizada por um nó de origem na rede, um nó de destino e um tamanho) é inserida na rede através de um caminho de custos marginais mínimos, em que deverão ser respeitados os seus prazos de entrega. Para tanto, é utilizado um método de caminhos mínimos com janelas de tempo, apresentado na Seção 4.1.

Esse processo é realizado demanda a demanda até que toda a rede esteja carregada, de forma que uma solução inicial é encontrada (Seção 4.2). Diferentes variantes desse método foram desenvolvidas, de forma a atender as restrições de caminhos únicos para cada par origem-destino de nós ou a possibilidade de ter variações ao longo do dia da semana, conforme apresentado na Seção 3.

Por fim, é também proposta uma heurística de busca local, em que as demandas são retiradas e reinseridas na rede, semelhante ao encontrado nos trabalhos de Hoppe *et al.* (1999), Erera *et al.* (2012) e Zhang (2010).

4.1 Caminhos mínimos com janelas de tempo

Para desenvolvimento da heurística construtiva no problema de roteirização de cargas fracionadas em uma rede de terminais, foi construído um algoritmo baseado em caminhos mínimos de janelas de tempo. Nessa Seção, será apresentado o método utilizado para determinar a rota de menor custo para cada demanda (caracterizada por uma origem, um destino e um tamanho) que respeita o seu respectivo prazo de entrega.

O algoritmo utilizado é semelhante ao observado em Desrochers *et al.* (1988a), Desrochers *et al.* (1986), Desrochers *et al.* (1988b) e Desrosiers (1988), se assemelhando também ao método de caminhos mínimos de Dijkstra descrito em

Ahuja et al. (1993). Nele, são associadas etiquetas (ou *labels*) aos diferentes nós da rede, mas, ao invés do método de Dijkstra, em que a única dimensão utilizada é o **custo**, também é adicionada uma variável relacionada ao tempo. As etiquetas terão formato (c, t), referente ao custo e ao instante de tempo associados ao nó, respectivamente.

Supondo um grafo direcionado *G(N,A)* com *N* nós e respectivos arcos *A,* temos especificados para cada par de nós *i* e *j* o custo de atravessar o arco, dado por c_{ij} ,bem como o tempo de transporte, t_{ij}.

Dada uma demanda *q* com origem or_q e destino ds_q, o algoritmo se inicia criando uma primeira etiqueta no nó de origem da demanda. Essa etiqueta deverá ter os valores (0, t_0), dado que o custo inicial é 0, e t_0 corresponde ao instante de criação da demanda, ou seja, o instante de tempo em que ela inicia sua jornada na rede.

A partir desse nó, deverão ser criadas etiquetas para todos os nós adjacentes, considerando os custos e tempos referentes a cada arco na rede, conforme observado na Figura 6.

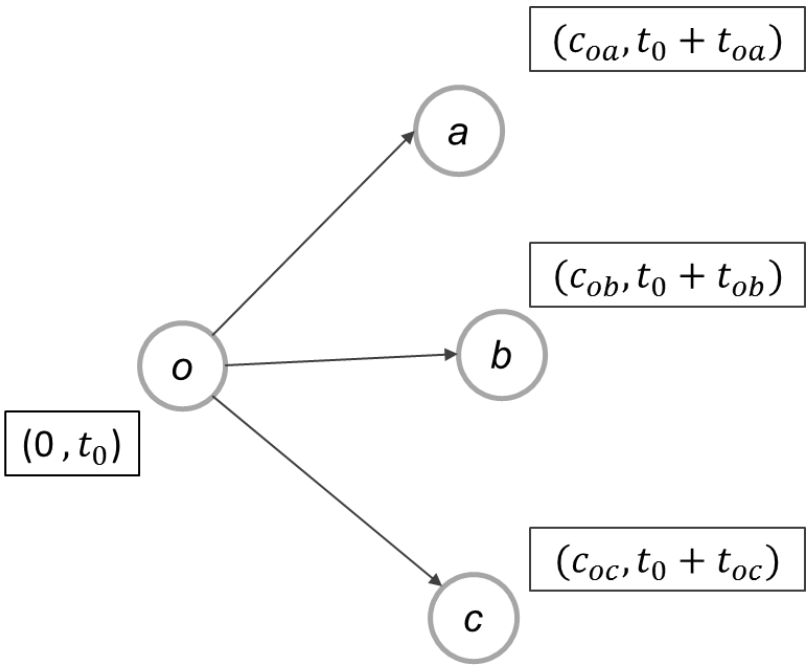

Figura 6: Exemplo de criação de etiquetas de nós adjacentes

Após a criação dos *labels* nos nós adjacentes, deverá ser realizada uma ordenação lexicográfica de todas as etiquetas identificadas na rede. Dessa maneira, terá sido obtida uma lista crescente das etiquetas de acordo com seu custo. Caso duas etiquetas tenham o mesmo custo, deverá ser ordenada primeiramente a com menor valor na dimensão de tempo.

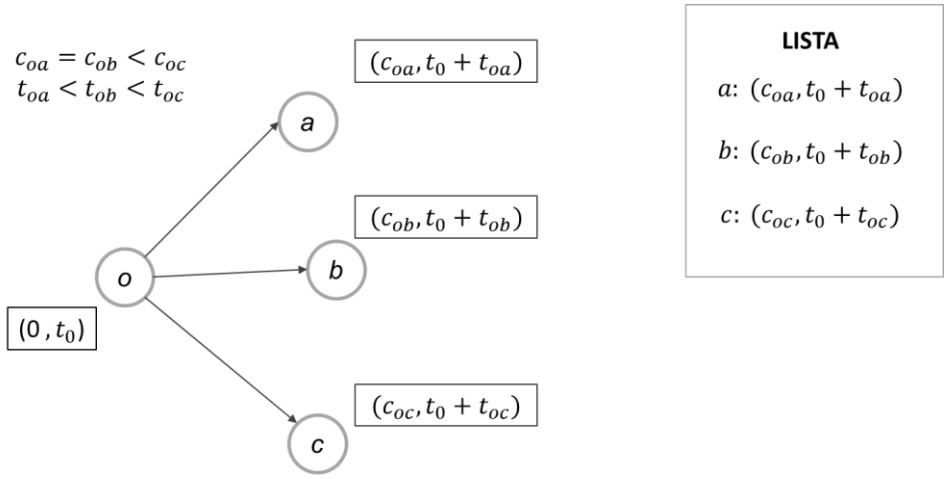

Figura 7: Exemplo de listagem de etiquetas

No exemplo da Figura 7, o primeiro *label* é o que consta no nó *a*. É importante ressaltar que, apesar de possuir o mesmo custo do *label* em *b* (c_{oa}=c_{ob}), ele é colocado primeiramente na lista, uma vez que a sua dimensão de tempo é menor (t_{oa}<t_{ob}).

Após a ordenação, deverá ser fixada a primeira etiqueta encontrada, em processo semelhante ao realizado em Dijkstra. Esse *label* (e seu respectivo nó) deverá ser então utilizado no próximo passo do algoritmo, em que novos *labels* serão criados a partir dos nós adjacentes.

Um aspecto relevante do algoritmo é a remoção das chamadas etiquetas dominadas de um nó. Supondo que existam dois *labels* diferentes associados a um mesmo nó, com valores (c_1, t_1) e (c_2, t_2). Caso c_1 seja menor que c_2 e t_1 seja maior que t_2, não se pode afirmar qual dos dois irá produzir um menor custo no destino final. Isso se dá pelo fato de que um instante de tempo anterior, embora com custo naquele momento maior, poderá posteriormente incorrer em reduções de custo no futuro.

Por outro lado, caso o custo e o instante de tempo da segunda etiqueta sejam maiores que os da primeira etiqueta, diz-se que esta está *dominada*, ou seja, caminhos que utilizem esse *label* não poderão produzir custos menores do que a outra alternativa. Nesse caso, a segunda etiqueta é removida da lista da rede, sendo mantidos, portanto, somente os *labels* não-dominados no prosseguimento do método.

Na Figura 8 é possível visualizar o que seria o próximo passo do algoritmo. A partir do *label* fixado no nó *a* são criados novos *labels* referentes aos nós adjacentes. Para o caso do nó *b*, o *label* formado foi considerado como *dominado* pelo *label* anterior, uma vez que o custo $c_{oa}+c_{ab}$ é maior ou igual a c_{ob}, e o termo ($t_0+t_{oa}+t_{ab}$) é maior do que ($t_0 + t_{ob}$), não sendo mantido, portanto.

Já para o nó *c*, podemos assumir que c_{oc} é maior do que o termo ($c_{oa}+c_{ac}$) e que o termo ($t_0+t_{oa}+t_{ac}$) é maior que ($t_0 + t_{oc}$). Nesse caso, nenhum dos dois *labels* é dominado pelo outro, que faz com que os dois tenham que ser mantidos nas próximas iterações.

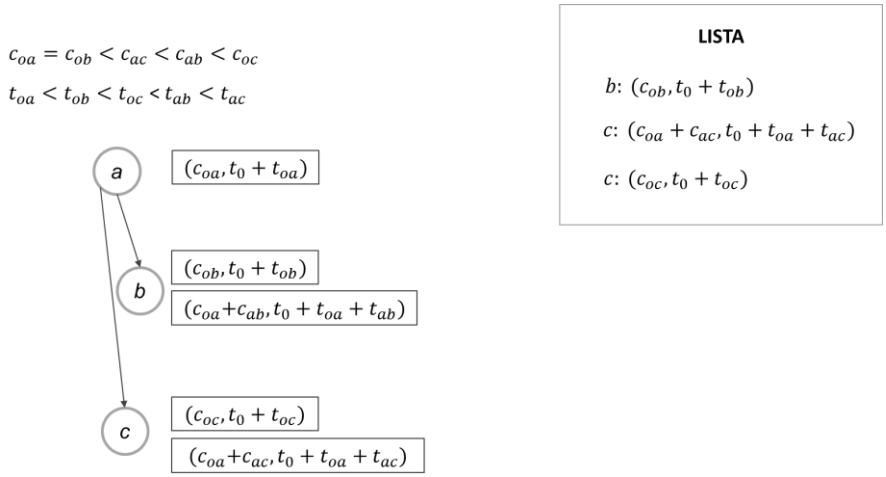

Figura 8: Exemplo de remoção de nós dominados

O algoritmo deverá prosseguir até que seja fixada uma etiqueta no nó de destino, caracterizando que se atingiu o menor custo possível. Um resumo em pseudo-código do algoritmo pode ser observado na Figura 11.

```
Caminho mínimo com janelas de tempo

label_atual = (0, t0)
Enquanto nó(label_atual) != destino
    Faça { cria_labels_adjacentes (label_atual)
           remove_dominados
           ordena_labels
           fixa_primeiro_label
           label_atual = primeiro_label

cria_labels_adjacentes (label_atual)

Para todos os nós adjacentes i de nó(label_atual)
    Faça { Adiciona_label (c0+c(0,i), t0+t(0,i))

remove_dominados

Para todos os nós da rede
    Faça { ordena_labels
           label_atual = primeiro_label
           Para todos os labels no nó
               Faça { Se(custo(label)>=custo(label_atual) e tempo(label)>=tempo(label_atual)
                      Então { remove label
```

Figura 9: Pseudo-código para caminho mínimo com janelas de tempo

4.2 Heurística construtiva

Conforme mencionado anteriormente, foram concebidas três variantes para o problema tratado, referente a três abordagens para a restrição de unicidade de caminhos: uma sem restrições, uma com caminhos únicos para todas as demandas que compartilhem um nó e tenham um mesmo destino, e uma terceira que permite flexibilizações de acordo com o dia da semana. Cada uma das variações do algoritmo proposto é apresentada nas seções 4.2.1, 4.2.2 e 4.2.3, respectivamente.

4.2.1 Algoritmo sem restrições de caminhos

Na primeira versão do algoritmo, todas as demandas serão carregadas separadamente na rede de terminais. Para isso, são considerados como parâmetro de entrada:

$\tilde{q}$: Vetor de demandas *q,* caracterizadas por um tamanho (s_q), uma origem (or_q), um destino (ds_q), um tempo de entrada no sistema (e_q) e um tempo máximo de entrega (p_q).

c_{ij}: Custo de transporte entre os terminais *i* e *j*

m_i: Custo de movimentação no terminal *i*

t_{ij}: Tempo de transporte entre os terminais *i* e *j*

$h_{i:}$ Tempo de movimentação no terminal *i*

Ao longo do algoritmo, as demandas serão alocadas em rotas na rede, fazendo com que sejam associados horários de chegada aos diferentes terminais. Dessa forma, podemos definir:

a_j^q= Horário de chegada da demanda *q* no terminal *j*

Outro objeto relevante no algoritmo são as chamadas **pernas de transporte**, caracterizadas por um veículo que atravessa um arco em um determinado horário, de maneira semelhante ao conceito de perna apresentado na Seção 3.1 e exemplificado na Figura 4. A cada perna poderão ser atribuídas diferentes demandas, que compartilham o transporte no arco percorrido. Cada perna *L* que atravessa um arco (*i, j*) possui os seguintes atributos:

Cap= Capacidade do veículo

d_i^l= Horário de saída da perna *l* do terminal *i*

a_j^l= Horário de chegada da perna *l* no terminal *j.* A diferença entre a_j^l e d_i^l deverá ser o tempo de transporte t_{ij}

Max_i^l= Máximo instante de saída possível do terminal *i*, caracterizado pelo máximo instante em que o veículo pode sair de *i* sem que nenhuma das demandas presentes na perna tenha seus prazos afetados

O conceito da perna é particularmente importante para o algoritmo, uma vez que demandas que sejam alocadas a uma mesma perna estarão, na prática, dividindo entre si os custos de transporte do arco. Além disso, elas deverão obrigatoriamente ter os mesmos instantes de saída do terminal *i* e de chegada no terminal *j*, ainda que suas rotas no restante da rede possam ser totalmente diferentes.

No procedimento construído, todas as demandas são ordenadas e inseridas uma a uma na rede, identificando para cada uma o seu caminho com menor custo marginal, utilizando o algoritmo de caminho mínimo com janelas de tempo (Seção 4.1).

Em especial, a ordenação se mostra um aspecto importante no algoritmo, uma vez que a ordem com que as demandas são inseridas irá definir quais sinergias o algoritmo poderá ser capaz de identificar ao longo da rede.

Uma primeira alternativa seria a ordenação simples decrescente em relação ao tamanho das demandas, de maneira semelhante ao realizado em Silva (2010). Intuitivamente, demandas com menor volume têm mais chances de conseguir ser "encaixadas" em pernas já carregadas na rede, devendo ser deixadas para o final do algoritmo, portanto.

Porém, foi proposta a inserção de uma aleatoriedade na ordenação de demandas, de maneira a gerar múltiplos casos aleatórios de ordenação de demandas, para em seguida realizar o processamento do algoritmo e verificar qual o melhor caso.

Dessa maneira, ao invés de uma ordenação simples pelo tamanho s_q de cada demanda *q*, optou-se por inserir um fator aleatório α que multiplica cada demanda, em uma lógica similar ao método GRASP. O resultado desse produto irá gerar um outro atributo, que será então utilizado para fazer a ordenação. Desse modo, mantém-se uma ordenação com estrutura decrescente de maneira geral, porém com um fator de aleatoriedade nela embutido. O cálculo do fator o_q utilizado para a ordenação é dado pela Equação (4.1).

$$o_q = s_q(1 + \alpha) \qquad (4.1)$$

Ao final, o algoritmo pode ser descrito conforme o pseudo-código apresentado na Figura 10.

Heurística Construtiva

Para a=1 **até** a=max_iteracoes

Faça
- Sorteia fator aleatório de ordenação o_q para cada demanda q
- Ordena vetor $\tilde{q}$ contendo demandas q a partir de o_q
- **Para cada** q do vetor $\tilde{q}$
  - **Faça**
    - Identifica rota *k* com custo mínimo entre origem e destino através do algoritmo de caminho mínimo com janelas de tempo
    - Aloca rota *k* na rede, criando e alterando pernas nos arcos

Figura 10: Pseudo-código para heurística construtiva

Especificamente para a identificação dos caminhos mínimos, é considerado o **custo marginal** na rede para cada demanda. Dessa maneira, caso uma demanda consiga compartilhar um veículo em determinado arco com uma outra demanda já previamente roteirizada (ou seja, que seja alocada a uma **perna** já existente), o seu custo marginal de transporte nesse arco será zero, e o algoritmo deverá considerar somente os custos de movimentação no terminal de destino.

Podemos supor que, no processo de construção de etiquetas na identificação de caminhos mínimos, o algoritmo esteja roteirizando uma certa demanda *q* em um dado nó *i*, que tenha um nó adjacente *j*, conforme exposto na Figura 11, tendo como destino final o nó *f* ($sd_q = f$).

O método deverá verificar se existe uma perna L* que já atravessa o arco (*i*,*j*) e que a demanda *q* possa compartilhar. Caso exista uma perna já utilizada por uma rota anterior em que a demanda avaliada possa ser alocada, deverá ser criado uma etiqueta no nó *j* em que o custo deverá ser somente o custo de movimentação nesse nó, sem acrescentar o custo de atravessar (*i*,*j*).

Além desse *label*, o algoritmo prevê também a criação de um outro, em que a demanda parte o mais cedo possível do nó *i* para o nó *j*, sem nenhum tipo de consolidação. Esse *label* deverá ter um custo maior ou igual ao anterior, bem como um tempo menor ou igual. A criação desse segundo *label* procura evitar situações em que a economia associada ao arco (*i,j*) não compense o tempo perdido para garantir o compartilhamento da perna L*, que poderia ser utilizado posteriormente em um outro arco mais vantajoso.

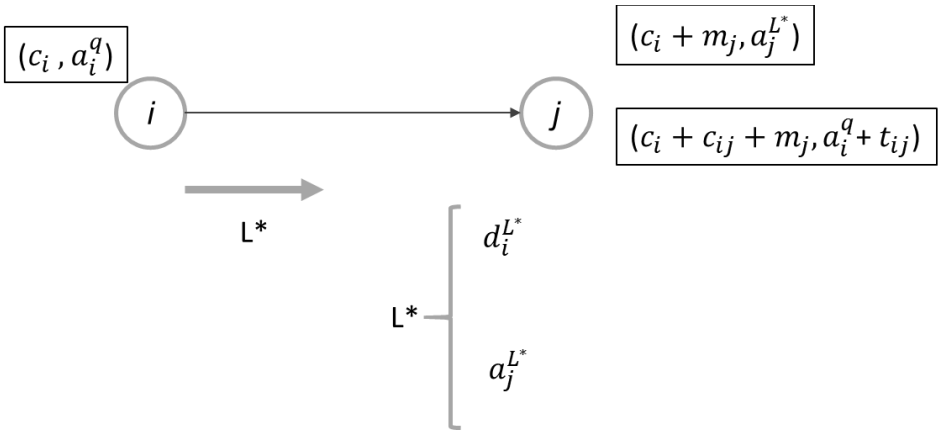

Figura 11: Cálculo de custo marginal

Dessa maneira, a fórmula de cálculo dos *labels* em *j* se dará da maneira apresentada na Figura 12.

```
Cálculo de custos de etiquetas

Para etiqueta analisada no nó l
Faça {
    //Cria label com consolidação
    Se existe perna no arco (i,j) e
        Há factibilidade temporal e
        Veículo tem capacidade
    Então {
        Cria novo label
        Custo_label (j) = Custo_label (i) +m(j)
        Tempo_label(j) = Cálculo_tempo_perna
    }
    //Cria label sem consolidação
    Cria novo label
    Custo_label(j)=Custo_label(i)+c(i,j)+m(j)
    Tempo_label(j)=Tempo_label(i)+t(i,j)+h(j)
}
```

Figura 12: Pseudo-código para cálculo de custos de etiquetas

Em especial para a identificação da factibilidade temporal de se realizar a consolidação de uma demanda com uma perna já existente, deverão ser verificados o horário previsto de chegada no nó *j* e o horário de máxima saída do nó *i.* O conceito de folgas que a perna pode ter, ou seja, o quanto ela pode ser deslocada temporalmente é semelhante ao proposto em Erera *et al.* (2008), em que é abordada a alocação de motoristas a uma rede de veículos.

Retornando ao exemplo anterior, podemos supor que uma demanda *r* já estivesse alocada à perna L*, tendo um prazo de entrega p_r no terminal *v*, e que após a passagem pelo arco (*i,j*), ela utilize o arco (*j,v)*, conforme visto na Figura 13. Nesse caso, a máxima saída do arco (*i,j*) deverá ser dada por:

$$Max_i^{L^*} = p_r - t_{jv} - h_j - t_{ij} \qquad (4.2)$$

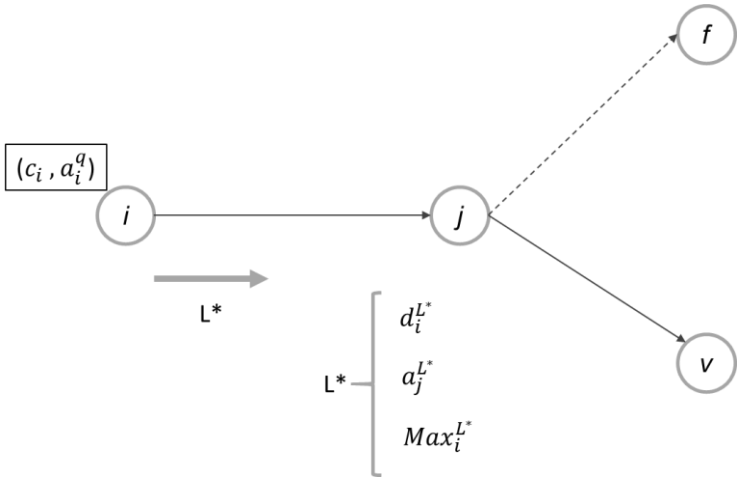

Figura 13: Verificação de factibilidade de consolidação

Ou seja, a máxima saída depende do prazo máximo de entrega e de todos os tempos intermediários. Caso existam outras demandas na perna avaliada, o instante de tempo máximo de saída em *i* deverá ser o mínimo entre o cálculo para cada uma.

Por outro lado, caso a demanda *q* tenha um prazo de entrega p_q, deverá ser verificado também se a consolidação com a perna *L** torna a entrega ainda factível. Para isso, deverá ser avaliado se o instante de chegada no terminal *j* não infactibiliza o prazo. No limite, a demanda poderia seguir diretamente de *j* para o seu destino final *f*. Ou seja, se ao instante de chegada em *j* forem somados o tempo de movimentação dentro do terminal e o tempo de transporte (*j,f*), que é o menor possível, e o prazo não seja cumprido, a consolidação com a perna é infactível. Isso pode ser colocado da seguinte forma:

$$\begin{cases} Se\ a_j^{L^*} + h_j + t_{jf} \leq p_q, consolidação\ é\ factível \\ Caso\ contrário, consolidação\ não\ é\ factível \end{cases}$$

Outro aspecto importante é que o instante de tempo esperado de chegada no terminal *j* deverá depender do instante de tempo atual de saída da perna do terminal *i* ($d_i^{L^*}$) e do instante de tempo previsto de chegada em *i* da demanda que está sendo avaliada (a_i^q).

Se o instante de tempo da etiqueta da demanda em *i* for anterior à saída da perna L* ($a_i^q \leq d_i^{L^*}$), não há mudança na perna, sendo equivalente à situação em que a

demanda “aguarda” no terminal a saída do seu veículo. Deverá ser considerado como o tempo de chegada em *j*, portanto, o valor $a_j^{L^*}$ já calculado previamente para a perna.

Já caso a saída prevista para a perna em *i* seja anterior ao instante de tempo de chegada da demanda em *i* ($a_i^q \geq d_i^{L^*}$), o instante de tempo previsto de chegada em *j* deverá ser recalculado, uma vez que a perna deverá atrasar a sua saída para que a demanda seja a ela alocada. Nesse caso, o seu cálculo seria dado por $a_j^q = a_i^q + t_{ij}$. Isso só é possível, naturalmente, caso o tempo do *label* da demanda avaliada no terminal *i* seja inferior ao tempo máximo de saída da perna em questão.

$$\begin{cases} Se\ a_i^q > d_i^{L^*}, \qquad d_i^{L^*} = a_i^q\ e\ a_j^{L^*} = a_i^q + t_{ij} \\ \quad Caso\ contrário, d_i^{L^*}\ e\ a_j^{L^*}\ se\ mantém \end{cases} \tag{4.3}$$

Dessa maneira, são realizadas duas checagens para se certificar de que temporalmente, a consolidação é possível, conforme apresentado no pseudo-código apresentado na Figura 14.

```
Checa_factibilidade (Label label_i, Perna L)

label_i //Label da demanda avaliada no nó i analisado
Perna L //Perna avaliada para consolidação no arco (i,j) com a demanda
t(i,j) //Tempo de translado entre os nós i e j na rede
t(j,d) //Tempo mínimo entre o nó j e o nó d de destino da demanda avaliada
Factibilidade =verdadeiro
Se label_i(tempo)>maxima_saida(L)
Então { Factibilidade =falso
Caso contrário
Então {
    Se label_i(tempo)>saída(L)
    Então { Instante_chegada(j) = label(i)+t(i,j)
    Caso contrário
    Então { Instante_chegada(j) = instante_chegada(L)
    Se instante_chegada(j)+t(j,d)>p_k
    Então { Factibilidade =falso
Retorna Factibilidade
```

Figura 14: Pseudo-código para checagem de factibilidade temporal de consolidação de demandas

O processo de criação dos *labels* deverá ser repetido até que o destino final da demanda *q* tenha sido encontrado. Ao final, uma rota terá sido identificada, podendo haver consolidações ou não nos diferentes arcos percorridos.

Nesse instante, a rota *k* deverá ser carregada na rede. Nesse processo, todos os arcos em que houve consolidação deverão ser revistos em termos de saída máxima, além de criadas novas pernas na rede. O algoritmo deverá se dar conforme o apresentado na Figura 15.

Carrega_demanda (rota$_k$)

Para cada par (*i,j*) na rota *k*

Então
- **Se** perna L*(*i,j*) é compartilhada
  - **Então**
    - Acrescenta demanda na perna L*
    - **Se** (label(i)>Saída(L*))
    - **Então**
      - Saída (L*) = label (i)
      - Chegada (j) = label(i)+t(i,j)
    - Atualiza tempo máximo de saída
- **Caso contrário**
  - **Então**
    - Cria perna L** no arco (*i,j*) com a demanda *k*

Figura 15: Pseudo código para carregamento de demanda na rede

Para exemplificar todo o algoritmo, podemos imaginar uma pequena rede de terminais com custos e tempos de transporte conforme a Figura 16. Para cada arco, são indicados os custos e tempos de transporte, respectivamente. Para efeitos de simplificação, todos os terminais possuem um tempo de processamento de 1 dia, com custos de movimentação zerados.

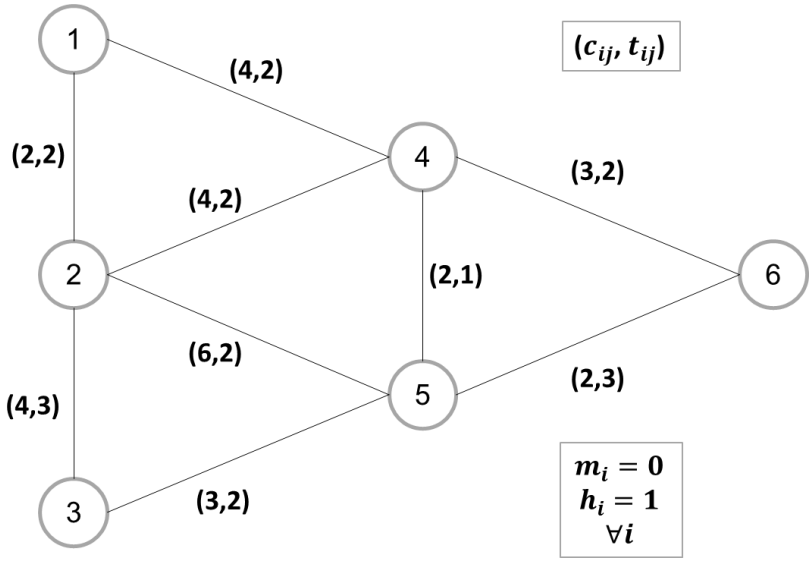

Figura 16: Pequena rede de terminais

No exemplo, suponhamos que existam três demandas para serem roteirizadas na rede, conforme apresentado na Tabela 2. A ordem em que deverá ser realizada a

alocação das demandas é exatamente a que consta na tabela, ou seja, supõe-se que a ordenação prevista utilizando a equação 4.1 já foi realizada.

Tabela 2: Exemplo de demandas para serem roteirizadas na rede

Demanda	Tamanho	Origem	Destino	Entrada no sistema	Prazo máximo de entrega
1	0,6	2	6	0	11
2	0,5	3	6	4	10
3	0,4	1	6	1	10

Na primeira etapa do algoritmo, a demanda 1 deverá ser roteirizada na rede, por enquanto vazia. O caminho percorrido será, portanto, o menor custo possível que ainda atenda os prazos. O resultado dessa iteração produz uma rota 2-4-6, vista em laranja na Figura 17. Deverão ser criadas duas pernas, L^1 e L^2 para os arcos (2,4) e (4,6), respectivamente.

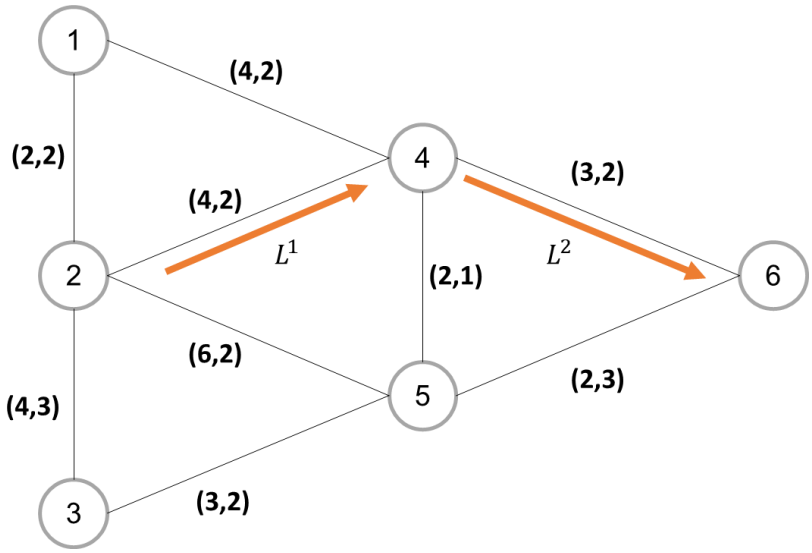

Figura 17: Rede após alocação da demanda 1

As duas pernas, no momento, deverão sair no primeiro instante possível. Mas, dado que o prazo da demanda 1 é no dia 11 e a chegada prevista é no dia 9, as duas pernas podem atrasar em dois dias as suas saídas. A relação das pernas criadas até essa etapa pode ser vista na Tabela 3.

Tabela 3: Relação de pernas na rede após alocação da demanda 1

Perna	Arco	Ocupação	Saída *i*	Chegada *j*	Máxima saída *i*
L^1	(2,4)	0,6	0	2	6
L^2	(4,6)	0,6	3	5	9

No próximo passo do algoritmo, a demanda 2 deverá ser roteirizada. Uma das possibilidades de redução de custo seria a consolidação com as pernas já criadas no sistema para a demanda 1. Porém, ao analisar a factibilidade temporal, essa consolidação se torna infactível. Isso se dá porque a demanda é inserida no sistema no instante de tempo 4. A sua chegada mínima no nó 2 para realizar a consolidação com a perna L^1 seria no instante 8 (considerando os 3 dias de transporte no arco entre 3 e 2 mais 1 dia de consolidação). Dado que a máxima saída da perna L^1 é no instante 5, não é possível realizar a consolidação, pois atrasaria a entrega da demanda 1.

Dessa maneira, a demanda 2 irá seguir pelo caminho de menor custo sem realizar consolidações. A sua rota 3-5-6 pode ser observada em laranja na Figura 18. Novamente, deverão ser criadas duas novas pernas no sistema, L^3 e L^4, para os arcos (3,5) e (5,6). A relação de todas as pernas para essa etapa do algoritmo pode ser vista na Tabela 4. Importante notar que a demanda 2, ao seguir o caminho 3-5-6, chega no seu destino exatamente no prazo máximo. Isso faz com que não haja margem para atraso na saída das pernas L^3 e L^4, ao contrário do visto nas pernas L^1 e L^2.

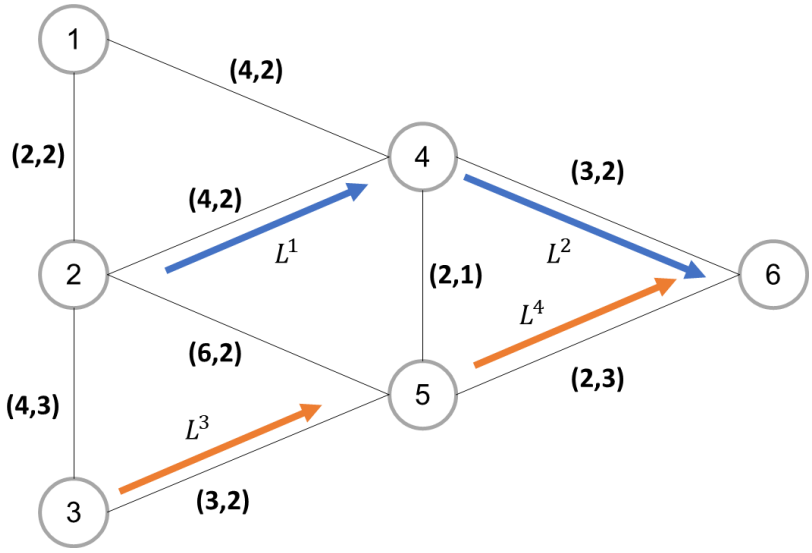

Figura 18: Rede após alocação da demanda 2

Tabela 4: Relação de pernas na rede após alocação da demanda 2

Perna	Arco	Ocupação	Saída *i*	Chegada *j*	Máxima saída *i*
L^1	(2,4)	0,6	0	2	6
L^2	(4,6)	0,6	3	5	9
L^3	(3,5)	0,5	4	6	4
L^4	(5,6)	0,5	7	10	7

Por fim, deverá ser roteirizada a demanda 3. Caso a rede não estivesse carregada, o seu caminho de custo mínimo seria 1-4-6, com custo total de 7. Porém, existe a possibilidade de consolidação com as pernas L^1 e L^2, fazendo com que o seu custo marginal esteja limitado à criação de uma nova perna no arco (1,2). Dessa forma, seu custo de inclusão no sistema seria de 2.

É importante analisar em detalhe a avaliação de factibilidade de consolidação da demanda 3 nas duas pernas. O primeiro ponto a ser avaliado é a sua capacidade. As duas pernas, nesse momento do algoritmo, contam com um volume total carregado de 0,6, ou seja, 60% da sua ocupação. Como a demanda 3 possui um volume de 0,4, a sua consolidação do ponto de vista de capacidade dos veículos é factível.

Do ponto de vista temporal, a demanda entra no sistema no instante 1. Isso faz com que ela esteja disponível para embarcar junto à perna no instante 4 (considerando 2 dias de transporte e 1 de movimentação dentro do terminal). Embora a perna L^1 saia

no instante 0, a sua máxima saída prevista é no instante 5. Ou seja, é possível atrasar a sua saída em 4 dias, permitindo a consolidação com a demanda 3. De maneira análoga, a consolidação com a perna L^2 também é factível.

Além disso, é necessário verificar se, do ponto de vista da demanda 3, o seu prazo ao consolidar com a perna L^1 pode ser cumprido. Ao sair do terminal 2 no instante 4, a demanda estaria pronta para um novo embarque no instante 7. Assumindo o caminho mais curto possível até o destino (ou seja, pegando imediatamente o arco entre 4 e 6), a chegada prevista seria no instante 9, menor que o prazo do dia 10. Dessa forma, também é factível a consolidação sob o aspecto do prazo da demanda a ser consolidada.

Dessa maneira, a configuração da rede após a alocação seria a vista na Figura 19, em que o caminho percorrido pela demanda 3 é demarcado em laranja.

Já a relação final de todas as pernas pode ser observada na Tabela 5. Além da criação de uma perna L^5, também foram realizadas modificações nas pernas L^1 e L^2 com a inserção da demanda 3. Além dos horários de saída terem atrasado, a máxima saída também foi alterada. Isso porque a demanda 3 chega no seu destino no instante 9, tendo seu prazo máximo o instante 10. Assim, embora a demanda 1, também presente nas pernas, permita que haja um atraso de dois dias, a demanda 3 faz com que o máximo atraso na saída das duas pernas seja de apenas 1 dia.

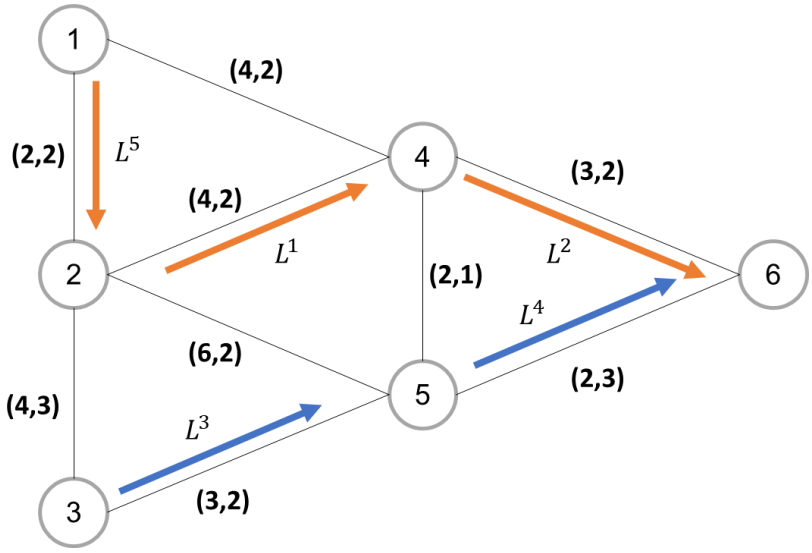

Figura 19: Rede após alocação da demanda 3

Tabela 5: Relação de pernas na rede após alocação da demanda 3

Perna	Arco	Ocupação	Saída *i*	Chegada *j*	Máxima saída *i*
L^1	(2,4)	1	4	6	5
L^2	(4,6)	1	7	9	8
L^3	(3,5)	0,5	4	6	4
L^4	(5,6)	0,5	7	10	7
L^5	(1,2)	0,4	1	3	2

4.2.2 Algoritmo com caminhos fixos

Conforme exposto na Seção 1, nas operações de carga fracionada, uma das restrições existentes é a de que os caminhos entre dois terminais na rede sejam únicos. Ou seja, caso duas demandas diferentes estejam em um determinado nó da rede e tenham o mesmo destino, elas devem seguir a mesma rota dali em diante, independentemente de quais as suas origens. Essa restrição faz com que os caminhos na rede tenham o formato de árvore, como exemplificado na Figura 20.

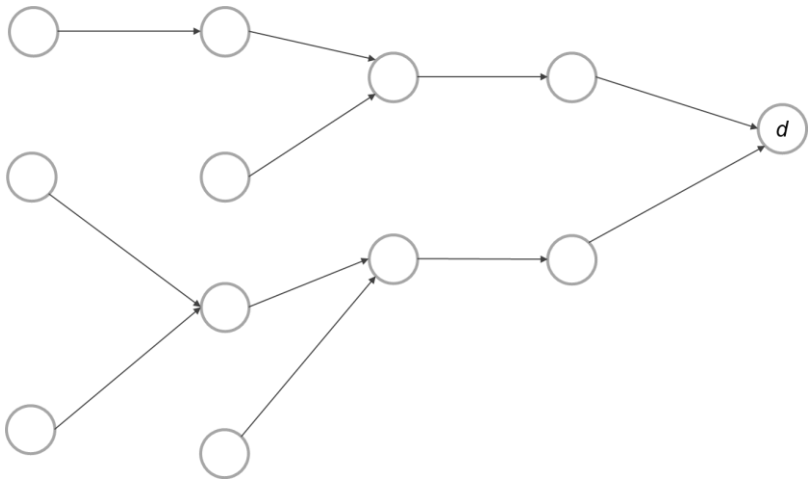

Figura 20: Exemplo de unicidade de caminhos para um destino *d* na rede

O método detalhado na Seção 4.2.1 não garante a unicidade dos caminhos, de forma que foi criada uma segunda variante para garantir que o formato de árvores dos caminhos seja respeitado. No caso, toda vez que uma demanda é inserida na rede, todas as demais inseridas posteriormente devem seguir o seu caminho, semelhante a um “rastro” deixado. Ou seja, toda demanda, ao verificar qual o caminho de menor custo até seu destino final, deve checar se já não existe um caminho pré-existente (ou “rastro”) deixado por uma demanda inserida em um momento anterior no algoritmo.

Para tanto, foi criado um verificador de caminho no algoritmo. No procedimento de identificação do caminho de menor custo, para cada *label* fixado, é checado se já existe um caminho pré-existente entre o nó atual e o destino final. Caso não tenha, a criação dos *labels* de nós adjacentes deverá se dar de maneira idêntica ao algoritmo descrito na Figura 9.

Caso já exista um caminho pré-determinado, ele é identificado e a busca de nós adjacentes para geração de *labels* se limita única e exclusivamente ao caminho especificado.

Imaginando que o algoritmo está analisando nós adjacentes de um nó *i* para uma demanda com destino a um nó *d* e foi identificado um caminho pré-definido *i-b-j-d*, só deverá ser considerada a criação de novas etiquetas para o nó *b*, ignorando, portanto, os outros nós adjacentes *a* e *c*.

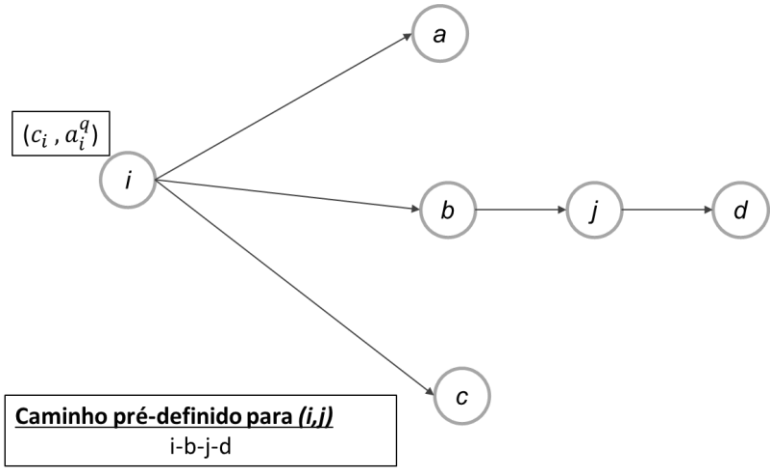

Figura 21: Exemplo de caminho pré-definido

Por fim, na alocação das rotas na rede, caso exista algum nó que não tenha um caminho pré-definido para o destino final, deverá ser criado esse rastro, para que todas as demandas no restante do algoritmo dali em diante sigam exatamente o mesmo percurso.

Dessa forma, a função de criação de *labels* adjacentes descrita na Figura 9 seria modificada, sendo atualizada de acordo com a Figura 22.

cria_labels_adjacentes (label_atual)

Identifica nó *i* referente ao *label* atual

Se existe caminho_pré-definido entre o nó *i* e o nó de destino *d*

Então
- Identifica caminho pré-definido entre o nó *i* e o nó *d*
- Encontra próximo nó *p* no caminho pré-definido
- Cria *labels* para o nó *p*

Caso contrário

Então
- **Para todos** os nós adjacentes *j* de nó(label_atual)
- **Faça** { Cria *labels* para *j*

Figura 22: Criação de *labels* para algoritmo com caminhos pré-definidos

Um aspecto relevante desse método é o de que as primeiras demandas irão definir os caminhos pré-definidos, afetando, portanto, as rotas que serão seguidas pelas demandas subsequentes. Isso pode causar, em última instância, na infactibilidade de

algumas rotas, caso a primeira demanda roteirizada tenha um prazo maior, gerando uma rota que outras demandas com prazos mais justos não conseguem acompanhar.

Para esse caso, gerou-se o conceito de **penalidade** na rede. Dessa maneira, no cálculo de custo entre dois *labels*, é incluída uma parcela relacionada ao não cumprimento do prazo, como pode ser visto na fórmula abaixo. Caso o caminho pré-determinado produza uma rota infactível, a demanda ainda assim deverá segui-lo, produzindo uma penalidade referente a quanto o prazo foi violado.

$$Penalidade = Custo_{\,penalidade} * Máximo\ (Chegada\ prevista\ destino - Prazo\ de\ entrega, 0)$$

Após todas as demandas terem sido roteirizadas, deverá ser realizada uma revisão daquelas em que foi encontrada uma penalidade. Primeiramente, deverá ser identificada aquela com maior penalidade e o respectivo nó na rota em que a infactibilidade foi produzida. A demanda então é marcada e removida da rede, bem como todas as demandas que utilizavam o mesmo caminho pré-determinado.

O caminho pré-determinado então é apagado e todas as demandas são roteirizadas novamente, iniciando pelas demandas com maior penalidade. Dessa maneira, as demandas com prazos mais apertados e maior chance de produzir penalidades serão alocadas na rede primeiro, definindo os caminhos pré-definidos e garantindo que os prazos de todas as subsequentes serão respeitados. Esse procedimento é realizado ao final da heurística construtiva, se estendendo também para também modificações na rede ocasionadas pelas heurísticas de melhoria (Seção 4.3.2). Um pseudo-código do algoritmo construtivo pode ser observado na Figura 23.

Heurística Construtiva

Para a=1 **até** a=max_iteracoes

Faça
- Sorteia fator aleatório de ordenação o_k para cada demanda q_k
- Ordena vetor $\tilde{q}$ contendo demandas q a partir de o_q
- **Para cada** q do vetor $\tilde{q}$
  - **Faça**
    - Identifica rota *k* com custo mínimo entre origem e destino da demanda q
    - Carrega rota *k* na rede
    - Carrega caminhos pré-definidos a partir da rota *k*
    - Verifica penalidade da demanda q
- **Enquanto** penalidades_rede>0
  - **Faça**
    - Identifica demandas *p* com penalidade
    - Retira da rede todas as demandas *v* que compartilham o par (nó, destino) causador de penalidade das demandas *p*
    - Reordena demandas *p* e *v* retiradas priorizando demandas com penalidade
    - Roteiriza cada demanda *v* e *p*

Figura 23: Pseudo código para heurística construtiva com caminhos fixos

Em relação ao código anterior apresentado na Seção 4.2.1, as principais diferenças estão relacionadas à criação de um caminho pré-definido a partir de uma rota criada e a inclusão de uma verificação de penalidade. Em especial, os caminhos pré-determinados são construídos conforme o código exibido na Figura 24.

Carrega caminhos pré-definidos (rotaK)

Para cada posição *n* na rota *k*

Faça
- Identifica nó i_n da posição *n* na rota *k*
- Vetor caminho_pre_definido (i_n,*d)* = [ ]
- **Para cada** posição *p* posterior a *n* (*p=n+1 ... d*)
  - **Faça**
    - Identifica nó i_p da posição *p* na rota *k*
    - Adiciona i_p a caminho_pre_definido (i_n,*d)*

Figura 24: Pseudo código para carregamento de caminhos pré-definidos

Retomando o exemplo da rede e demandas apresentadas na Figura 16, podemos verificar as diferenças em relação ao algoritmo anterior. Nesse exemplo, porém a

demanda 3 tem um volume de 0,5. A nova relação das demandas pode ser vista na Tabela 6.

Tabela 6: Exemplo de demandas para serem roteirizadas na rede para abordagem de caminhos fixos

Demanda	Tamanho	Origem	Destino	Entrada no sistema	Prazo máximo de entrega
1	0,6	2	6	0	11
2	0,5	3	6	4	10
3	0,5	1	6	1	10

Dado que as demandas 1 e 2 percorrem caminhos totalmente independentes na abordagem anterior, não haveria diferença nas suas rotas no novo algoritmo. Dessa maneira, ao roteirizar a demanda 3, a rede estaria carregada conforme a Figura 25, em que a relação de pernas é observada na Tabela 7.

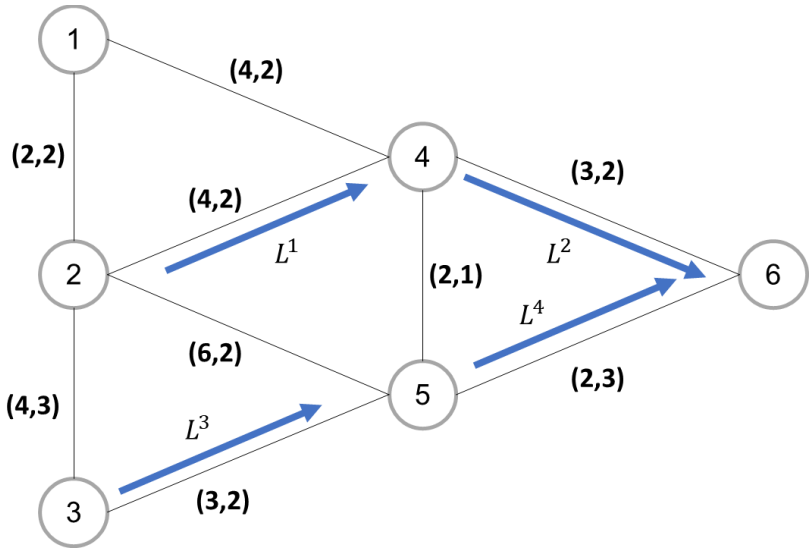

Figura 25: Configuração da rede anterior à alocação da demanda 3 para abordagem de caminhos fixos

Tabela 7: Relação de pernas anterior à alocação da demanda 3 para abordagem de caminhos fixos

Perna	Arco	Ocupação	Saída *i*	Chegada *j*	Máxima saída *i*
L^1	(2,4)	0,6	0	2	6
L^2	(4,6)	0,6	3	5	9
L^3	(3,5)	0,5	4	6	4
L^4	(5,6)	0,5	7	10	7

Com o novo tamanho da demanda 3, porém, não é possível realizar a consolidação com as pernas L^1 e L^2 devido à capacidade dos veículos. Caso o algoritmo não tivesse restrições de caminhos pré-definidos, o caminho percorrido seria o observado na Figura 26, em que a demanda se utilizaria da perna L^4. O seu custo marginal de inserção seria de 6, estando associado à criação das pernas L^5 e L^6, conforme visto na tabela Tabela 8.

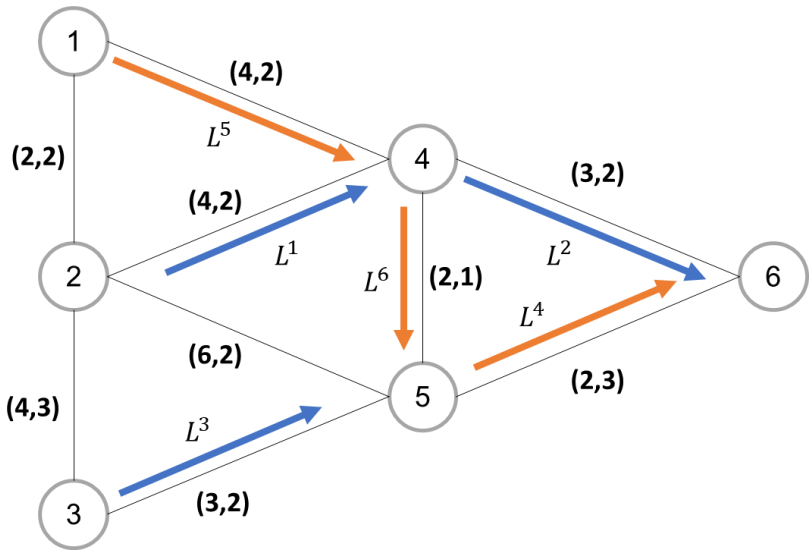

Figura 26: Configuração da rede após alocação da demanda 3 sem restrições

Tabela 8: Relação de pernas após alocação da demanda 3 sem restrições

Perna	Arco	Ocupação	Saída *i*	Chegada *j*	Máxima saída *i*
L^1	(2,4)	0,6	0	2	6
L^2	(4,6)	0,6	3	5	9
L^3	(3,5)	0,5	4	6	4
L^4	(5,6)	1	7	10	7
L^5	(4,2)	0,5	1	3	1
L^6	(4,5)	0,5	4	5	4

Porém, ao chegar no nó 4, já existe um caminho pré-definido de rota para demandas com destino ao nó 6, criado com a inserção da demanda 1. Isso faz com que obrigatoriamente a demanda 3 tenha que seguir esse percurso, mesmo que a consolidação com a perna L^4 traga maiores economias. Dado que não é possível a consolidação com a perna L^2, deverá ser criada uma nova perna, L^6, no mesmo arco. Ao final, o custo marginal de inserção da demanda 3 seria de 7. A configuração da

rede seria a mostrada na Figura 27, e a relação de pernas o apresentado na Tabela 9.

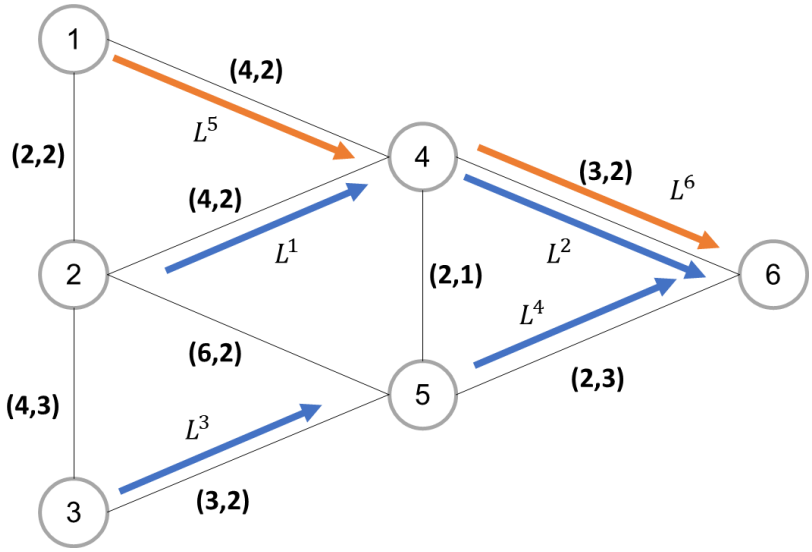

Figura 27: Configuração da rede após alocação da demanda 3 com abordagem de caminhos fixos

Tabela 9: Relação de pernas após alocação da demanda 3 com abordagem de caminhos fixos

Perna	Arco	Ocupação	Saída *i*	Chegada *j*	Máxima saída *i*
L^1	(2,4)	0,6	0	2	6
L^2	(4,6)	0,6	3	5	9
L^3	(3,5)	0,5	4	6	4
L^4	(5,6)	0,5	7	10	7
L^5	(4,2)	0,5	1	3	5
L^6	(4,6)	0,5	4	6	8

4.2.3 Algoritmo com caminhos por dia da semana

Para atender à restrição de unicidade de caminhos, além da variante de caminhos pré-determinados para pares origem-destino, foi também criada uma outra versão do algoritmo, em que também são definidos caminhos fixos para dois pares de nós, porém esses caminhos podem variar de acordo com o dia da semana.

Essa variação da restrição de caminhos em árvores já foi tratada nos trabalhos de Erera *et al.* (2012). De acordo com os autores, com os avanços tecnológicos em termos de rastreamento de mercadorias nos terminais, esse tipo de flexibilização seria factível operacionalmente, produzindo redução nos custos totais da rede.

A estratégia utilizada para esse cenário é semelhante à utilizada na segunda variante, em que ao verificar as rotas de cada demanda, caso já exista um caminho pré-determinado para aquele dia da semana e o nó analisado e o nó de destino, só deverão ser verificados nós subsequentes dentro do caminho pré-determinado. A criação de *labels* no algoritmo, portanto, segue de maneira análoga ao pseudo-código apresentado na Figura 28.

cria_labels_adjacentes (label_atual)

Identifica nó *i* referente ao *label* atual

Identifica dia da semana *s* referente ao *label* atual

Se existe caminho_pré-definido entre o nó *i* e o nó de destino *d* no dia de semana *s*

Então
- Identifica caminho pré-definido entre o nó *i* e o nó *d* no dia de semana *s*
- Encontra próximo nó *p* no caminho pré-definido
- Cria *labels* para o nó *p*

Caso contrário

Então
- **Para todos** os nós adjacentes *j* de nó(label_atual)
- **Faça**
  - Cria *labels* para *j*

Figura 28: Criação de *labels* para caminhos por dia da semana

A heurística construtiva se dará, de maneira geral, de modo bastante semelhante ao algoritmo de caminhos fixos, em que deverá ser sempre feita ao final uma verificação de penalidades e eventuais retiradas e reinserções de demandas na rede. O pseudo-código pode ser observado na Figura 29.

Heurística Construtiva

Para a=1 **até** a=max_iteracoes

Faça
- Sorteia fator aleatório de ordenação o_q para cada demanda q
- Ordena vetor $\tilde{q}$ contendo demandas q a partir de o_q
- **Para cada** q do vetor $\tilde{q}$
  - **Faça**
    - Identifica rota *k* com custo mínimo entre origem e destino
    - Carrega rota *k* na rede
    - Carrega caminhos pré-definidos semanais a partir da rota *k*
    - Verifica penalidade da demanda q
- **Enquanto** penalidades_rede>0
  - **Faça**
    - Identifica demandas *p* com penalidade
    - Retira da rede todas as demandas *v* que compartilham o par (nó, destino) e dia de semana causadores de penalidade das demandas *p*
    - Reordena demandas *p* e *v* retiradas priorizando demandas com penalidade
    - Roteiriza cada demanda *v* e *p*

Figura 29: Pseudo código da heurística construtiva com caminhos por dia da semana

Também de maneira similar ao algoritmo de caminhos fixos, a inserção de demandas na rede, após identificação do caminho de menor custo, também requer uma adaptação referente à criação dos caminhos pré-determinados na semana, como pode ser visto no pseudo-código da Figura 30.

Carrega caminhos pré-definidos semanais (rotak)

Para cada posição *n* na rota *k*

Faça
- Identifica nó i_n da posição *n* na rota *k*
- Identifica dia da semana *s* no nó i_n da posição *n* da rota *k*
- Vetor caminho_pre_definido (i_n,*d, s)* = [ ]
- **Para cada** posição *p* posterior a *n* (*p=n+1 ... d*)
  - **Faça**
    - Identifica nó i_p da posição *p* na rota *k*
    - caminho_pre_definido (i_n,*d, s)* = caminho_pre_definido (i_n,*d, s)* + i_p

Figura 30: Pseudo código do carregamento de caminhos pré-definidos por dia da semana

Existe, porém, uma maior complexidade para esse modelo, uma vez que uma alteração temporal pode modificar a rota que uma demanda deveria seguir. Isso é especialmente crítico nas avaliações de consolidação de uma demanda com uma perna existente.

Como visto no algoritmo apresentado na Seção 4.2.1, a inclusão de uma demanda em uma perna pode levar a uma postergação da sua saída (desde que respeite a saída máxima permitida). Essa postergação, porém, pode fazer com que o dia da semana da chegada em outros pontos da rota das demandas presentes na perna possam ser alterados. Consequentemente, as rotas previstas anteriormente estariam classificadas como infactíveis.

Imaginemos que uma demanda *r* tem como origem um nó *i* e destino final *v*. Para isso, são utilizados dois arcos, (*i,j*) e *(j,v)*. Uma demanda *q* de destino *f,* com *label* no nó i, está passando pelo processo de avaliação de consolidação com a perna L*, utilizada pela demanda *r* para atravessar o arco (*i,j),* conforme observado na Figura 31.

Na situação atual, a perna L* sai numa segunda-feira do nó *i* e chega na terça-feira em *j,* porém o prazo de entrega de *r* permite que a partida seja feita até terça-feira. Caso o *label* da demanda *q* em *i* seja na terça-feira, a consolidação faria com que a demanda *r* chegasse na quarta-feira, e não mais na terça-feira. A troca do dia de semana que chega pode gerar uma situação com um novo caminho pré-determinado, alterando o percurso que *r* deveria seguir de *j* até seu destino final *v*.

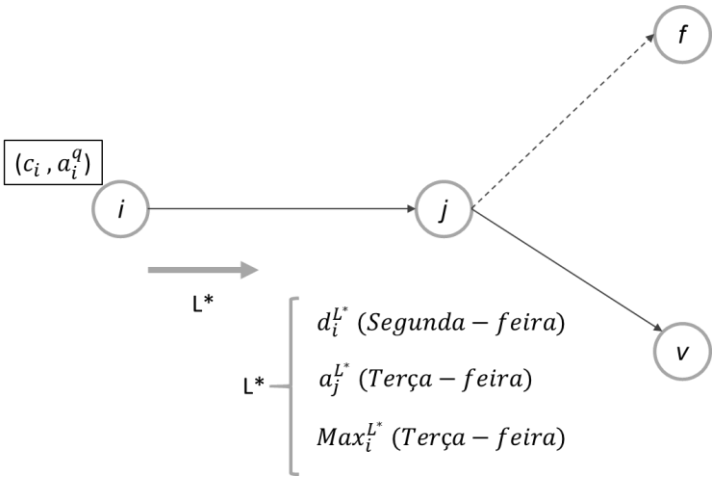

Figura 31: Exemplo de consolidação com mudança no dia da semana

No algoritmo, portanto, para cada consolidação realizada, é feita uma verificação para todas as demandas presentes na perna para se certificar de que não houve uma violação em termos de dia da semana e rotas pré-definidas. Caso seja identificada

alguma inconsistência, a demanda que teve a sua rota alterada será retirada da rede para ser roteirizada novamente ao final.

Dado que a verificação de consolidação se torna significativamente mais complexa, bem como há um maior número de casos de demandas que precisam ser roteirizadas novamente, esse método por vezes exige um maior tempo de processamento. Isso pode ser considerado contra-intuitivo, dado que ele é mais restrito que o primeiro método.

Novamente, resgatando o exemplo da rede descrita na Figura 16, o resultado obtido com os caminhos fixos mostrado na Figura 27 não se repetiria com a abordagem de caminhos por dia de semana. Isso porque a demanda 3 chegaria no nó 4 no dia 5, enquanto o caminho pré-definido formado pela demanda 1 valeria somente para demandas que chegam no dia 4. Dessa maneira, o caminho percorrido pela demanda 3 seria o descrito na Figura 26.

Caso o instante de entrada do sistema da demanda 3 fosse 2, porém, a demanda chegaria no dia 4 no nó 4, fazendo com que o caminho pré-definido devesse ser respeitado, voltando à configuração prevista na Figura 27.

4.3 Melhorias locais

Além de um algoritmo para construção de uma solução inicial, também foi construída uma heurística de buscas locais visando melhorias a partir de uma dada solução. O processo, de maneira geral, consiste em retirar uma demanda ou um grupo de demandas da rede e realizar uma reinserção, buscando caminhos mais econômicos que, no momento da construção anterior, não eram disponíveis para avaliação. Esse procedimento visa atenuar o efeito da ordenação das demandas no início do procedimento construtivo, ou seja, uma determinada remessa poderia ser consolidada com uma outra que ainda não foi inserida no sistema, mas, no instante da avaliação de custos marginais, acaba percorrendo um caminho com custo marginal maior.

Procedimentos semelhantes de melhorias locais, em que há retirada e reinserção de demandas na rede, são implementados em diferentes trabalhos de planejamento de

redes de transporte fracionado, como Hoppe *et al.* (1999), Erera *et al.* (2012) e Zhang (2010).

De forma semelhante ao algoritmo construtivo, também foram geradas três variações do modelo de melhorias locais: um que não considera restrições de unicidade de caminhos, um que considera um caminho único para cada par origem-destino e um que permite uma flexibilização de acordo com o dia da semana.

4.3.1 Algoritmo sem restrições de caminhos

Na versão da heurística de melhorias sem necessidade de unicidade de caminhos, após a construção de uma solução inicial, é feita uma passagem uma a uma de cada demanda, em que é feita uma retirada da rede e uma nova inserção.

Esse procedimento permite que, para cada remessa, seja feita uma avaliação com a rede totalmente carregada, ou seja, com uma maior possibilidade de consolidações no transporte. Dessa maneira, sinergias com demandas inseridas em instantes diferentes na rede passam a ser possíveis.

Como exemplo, conforme exposto na Figura 32, podemos imaginar uma demanda q_1 que possui origem em um nó *o* e destino em um nó *d*, além de possuir um tamanho de 40% de um veículo. Os custos de cada arco são colocados como *c.*

Supondo que, no momento que essa demanda será avaliada para inserção na rede, exista somente uma outra demanda, que passa pelos arcos (*j,k*) e (*k, d)* e ocupa 50% do veículo. Por simplificação, supõe-se que temporalmente a consolidação é plenamente possível e que os custos de movimentação nos terminais são insignificantes frente aos custos de transporte.

Nesse caso, a demanda q_1 deverá percorrer o caminho *o-j-k-d,* uma vez que isso implicará somente em um custo marginal de abrir o arco (*o,j*), portanto de 6.

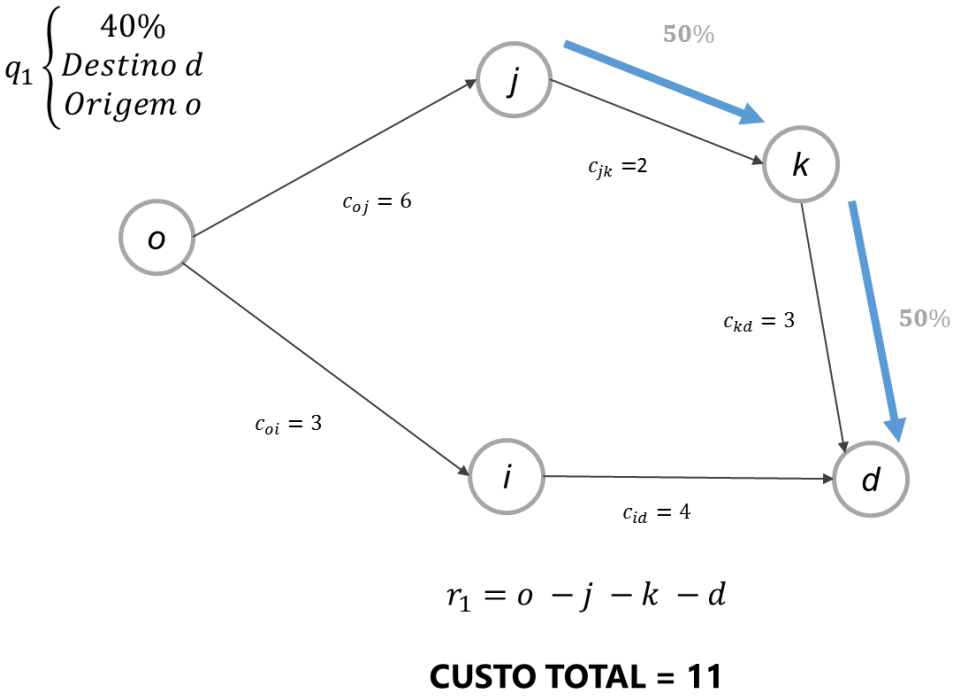

Figura 32: Exemplo de melhoria local

Imaginando que, num passo posterior, fosse inserida uma demanda q_2, com o mesmo destino e que passa pelo nó *o*, preenchendo 20% do caminhão. Dado o seu tamanho, não é possível realizar uma consolidação nos arcos (*j,k*) e (*k, d)*. Dessa maneira, deverá ser aberto um novo caminho *o-i-k*, conforme observado na Figura 33.

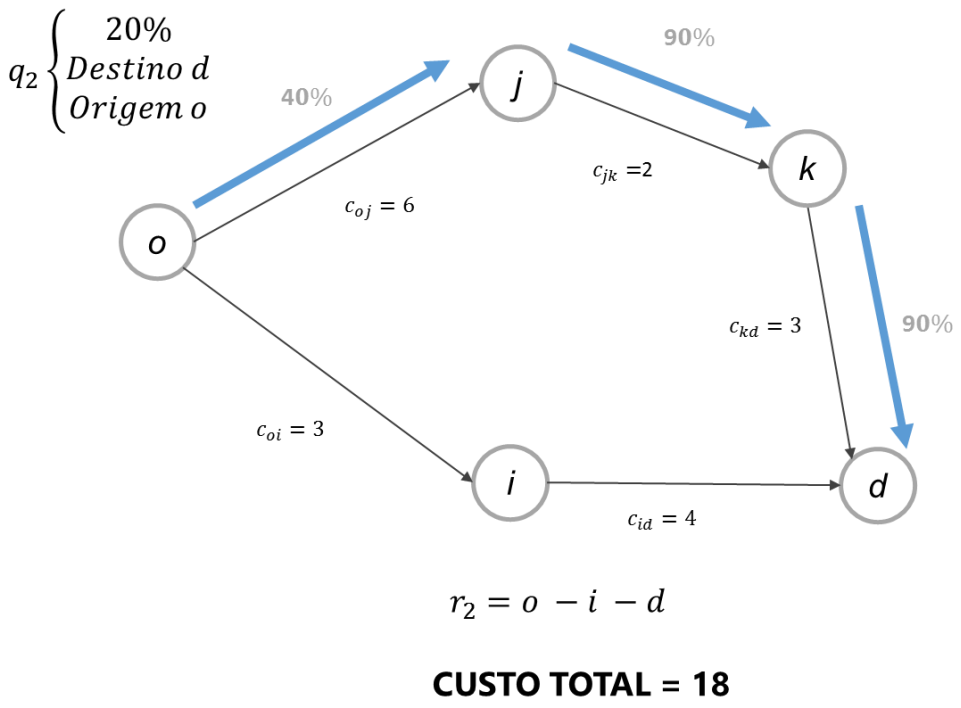

Figura 33: Exemplo de melhoria local

Nesse exemplo, é importante notar que, apesar dos custos marginais de cada demanda terem sido minimizados, isso não refletiu no custo global. Isso porque uma consolidação entre as demandas q_1 e q_2 implicaria numa melhor solução, em especial pela subutilização dos arcos percorridos pela segunda.

No algoritmo de melhoria, a demanda q_1 em um certo ponto é revisitada, devendo ser retirada e reinserida na rede. Nesse caso, como a demanda q_2 já está carregada na rede, o algoritmo tem visibilidade sobre essa sinergia e q_1 é redirecionada para um caminho que diminui os custos totais da rede (Figura 34).

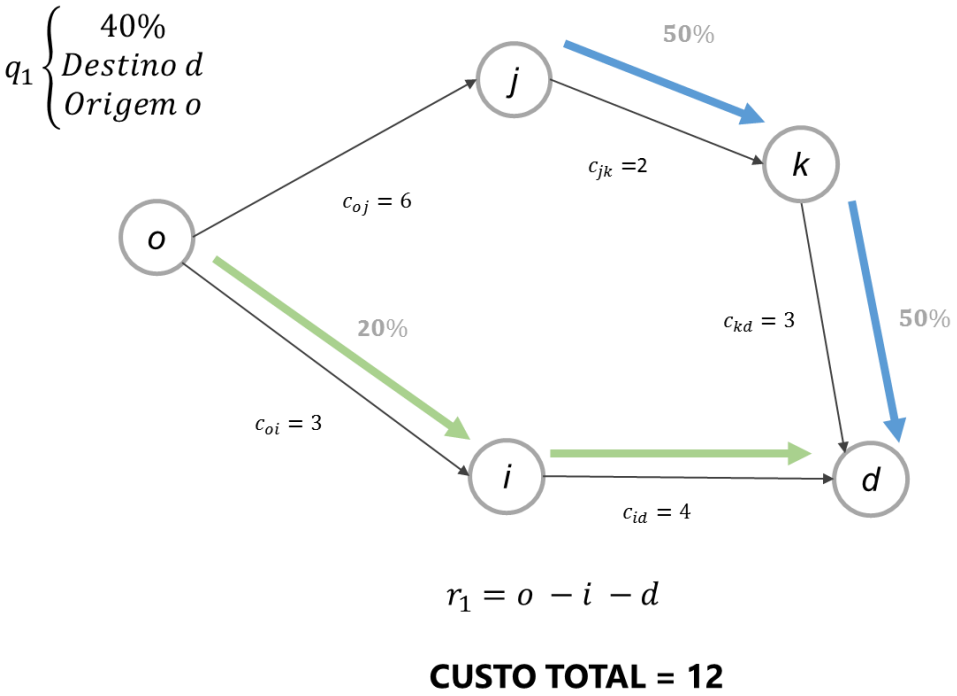

Figura 34: Exemplo de melhorias locais

O algoritmo de melhorias deverá produzir resultados mais significativos na eliminação de arcos em que somente a demanda analisada ocupa um arco. Nesse caso, o ganho do sistema com a eliminação do arco é imediato. Caso a demanda sempre compartilhe arcos no seu caminho, no processo de reavaliação, a inserção no seu caminho anterior deverá produzir custos de transporte sempre zero, e a comparação com caminhos alternativos deverá considerar somente os custos de movimentação. A rigor, não haveria necessidade de se gravar a solução anterior antes de fazer a reinserção de uma determinada demanda pois, na pior das hipóteses, ela deverá percorrer exatamente o mesmo caminho percorrido anteriormente, que faz com que a nova solução seja sempre melhor ou igual à anterior.

O algoritmo de reinserção de cada demanda pode ser processado repetidamente para todas as demandas da rede, uma vez que, a cada mudança na rota percorrida por uma demanda, novas oportunidades de sinergia que ainda não foram avaliadas podem surgir. Dessa maneira, o algoritmo proposto consiste de vários processamentos, até que um determinado número de iterações seja encontrado ou que um critério de parada por melhoria seja atingido.

Um pseudo-código do método utilizado pode ser observado na Figura 35.

```
Melhorias_locais sem restrições

Para cada demanda q carregada na rede
Faça {
    solução_anterior=solução_atual
    custo_inicial = custo (solução_atual)
    Retirar demanda q
    Roteirizar demanda q
    Se custo_inicial>custo (solução_atual)
    Então { Consolida solução_atual
    Caso contrário
    Então { Retorna solução_anterior
}
```

Figura 35: Pseudo código para melhorias locais sem restrições de unicidade

4.3.2 Algoritmo com caminhos fixos

Para o caso de melhorias locais com unicidades de caminhos, a retirada e reinserção de uma única demanda não produziria os resultados desejados, dado que os caminhos pré-definidos permaneceriam e a demanda acabaria, inevitavelmente, percorrendo o mesmo caminho anterior.

Portanto, ao avaliar uma rede carregada com a restrição de unicidade de caminhos, deverão ser selecionados dois **nós** da rede, e removidas **todas** as demandas que percorrem o primeiro com destino ao segundo. Dessa maneira, o parâmetro que indica qual o caminho que deve ser percorrido para esse par de nós pode ser eliminado e a roteirização pode ocorrer mais livremente.

Ainda assim, a nova roteirização de cada demanda estará sujeita a outros parâmetros que determinam rotas fixas. Dessa maneira, é possível que uma demanda, no processo de roteirização, encontre um nó que possui um caminho pré-determinado de demandas que não foram removidas da rede. Nesse caso, ela deverá seguir o caminho pré-determinado dali em diante, o que restringe o espaço de busca.

Uma maneira de diminuir a rigidez do problema e aumentar o potencial ganho em uma iteração seria o de remover todas as demandas que compartilham arcos com as demandas retiradas e seus respectivos caminhos pré-definidos, e todas as demandas e caminhos pré-definidos que compartilham arcos com essas demandas e assim por diante. Isso ocasionaria, porém, em praticamente uma nova roteirização de todo o sistema, fugindo da ideia de uma melhoria local.

Outro aspecto relevante é o de ordenação das demandas retiradas da rede para reinserção. Como na heurística construtiva, esse é um fator determinante para o resultado final da avaliação. No presente trabalho, de forma semelhante ao aplicado na geração de solução inicial, foi realizada uma ordenação em ordem decrescente de tamanho considerando um fator de aleatoriedade.

De forma semelhante também à heurística construtiva, é possível que uma demanda que tenha sido roteirizada no começo do algoritmo produza um caminho pré-determinado infactível temporalmente para uma demanda roteirizada posteriormente. Nesses casos, de forma semelhante ao descrito na Seção 4.2.2, foi utilizado o conceito de penalidades para caso de não cumprimento de prazos. Ao final, caso constatadas penalidades, todas as demandas serão retiradas novamente da rede e reinseridas, agora com as remessas que apresentavam violação de prazo sendo roteirizadas primeiramente.

Um pseudo-código do processo pode ser observado na Figura 36.

Melhorias locais com caminhos fixos

```
solução_anterior=solução_atual
custo_inicial = custo(solução_atual)
Para cada nó i da rede
Faça { Para cada nó j da rede
       Faça { Retira todas as demandas q que passam por i com destino a j
              Reordenada as demandas retiradas
              Roteiriza cada demanda q
Enquanto penalidade_rede>0
Faça { Identifica demandas p com penalidades
       Retira todas as demandas v que compartilham o nó com penalidade e o destino das demandas p
       Reordena as demandas p e v retiradas priorizando demandas com penalidade
       Roteiriza cada demanda v e p
Se custo_inicial> custo(solução_atual)
Então { Consolida solução_atual
Caso contrário
Então { Retorna solução_anterior
```

Figura 36: Pseudo código para melhorias locais para caminhos fixos

4.3.3 Algoritmo com caminhos por dia da semana

Para o caso de caminho fixos por dia da semana para cada par origem e destino, a simples retirada e reinserção de cada demanda, de modo semelhante ao descrito em 4.3.1, também não é possível. Isso se dá devido à presença dos parâmetros que indicam os caminhos que devem ser percorridos, de forma que a remessa retirada obrigatoriamente acabaria seguindo o caminho anterior.

Dessa forma, de maneira similar ao visto no algoritmo de caminhos fixos, é necessário selecionar um nó de passagem na rede, um nó de destino da rede e um dia da semana. A partir dessa seleção, todas as demandas que se utilizam desse nó nesse dia da semana nas suas rotas com destino ao outro nó deverão ser removidas para serem reinseridas novamente no sistema.

De forma semelhante ao método da Seção anterior, as demandas roteirizadas ainda estão sujeitas a outros parâmetros de rota pré-definida na rede, que faz com que haja

uma maior rigidez no sistema, diminuindo as possibilidades de ganho na melhoria local.

Outro aspecto também presente é a ordenação das demandas e a possibilidade de uma demanda roteirizada no começo do algoritmo produzir um caminho pré-determinado infactível para demandas subsequentes. Nesse caso, é utilizado novamente o conceito de penalidades. Ao final, todas as demandas com penalidades ganharão prioridade, e todas as demanda serão removidas e reinseridas na rede, agora com uma nova ordenação.

Na nova roteirização das demandas, é possível que sejam encontradas situações em que a consolidação de uma nova demanda em uma perna ocasione em mudança no dia da semana da sua chegada, gerando inconsistências nos caminhos pré-determinados, conforme descrito no final da Seção 4.2.3. Nesse caso, o mesmo procedimento é utilizado, em que demandas que tenham o seu dia de semana de chegada alterado são retiradas da rede para serem roteirizadas novamente posteriormente.

O pseudo-código do procedimento de melhorias locais com caminhos pré-determinados de acordo com o dia da semana é apresentado na Figura 37.

<u>**Melhorias locais com caminhos por dia da semana**</u>

solução_anterior=solução_atual
Custo_inicial = custo(solução_atual)
Para cada nó *i* da rede
Faça
- **Para cada** nó *j* da rede
  Faça
  - **Para cada** dia da semana *s*
    Faça
    - Retira todas as demandas *q* que passam por *i* no dia *s* com destino a *j*
    - Reordenada as demandas retiradas
    - Roteiriza cada demanda *q*

Enquanto penalidade_rede>0
Faça
- Identifica demandas *p* com penalidades
- Retira todas as demandas *v* que compartilham o instante de chegada e destino do nó com penalidade das demandas *p*
- Reordena as demandas *p* e *v* retiradas priorizando demandas com penalidade
- Roteiriza cada demanda *v* e *p*

Se custo_inicial> custo (solução_atual)
Então
- Consolida solução_atual

Caso contrário
Então
- Retorna solução_anterior

Figura 37: Heurística de melhoria para caminhos por dia da semana

5 RESULTADOS

Uma vez propostas uma modelagem matemática em programação linear inteira mista (Seção 3) e heurísticas construtivas e de melhoria local (Seção 4), foram realizados testes de implantação para diferentes casos.

Tais testes permitem que seja verificada a aplicabilidade dos métodos de solução propostos, tanto do ponto de vista de potencial de redução de custos quanto de tempos de processamento e factibilidade dos resultados.

Primeiramente, foram concebidas instâncias de pequeno porte com dados virtuais, gerados de maneira aleatória. O uso dessas instâncias permite que a programação linear inteira mista consiga ser aplicada, tornando possível a comparação de resultados com as heurísticas propostas. Dessa maneira, tem-se uma visão de qual seria a diferença em termos de função objetivo entre as soluções, bem como uma avaliação de qual a diferença em termos de tempo de processamento, fundamental para garantir que casos reais de maior porte sejam processados em tempo hábil.

Além disso, o método heurístico também foi processado com dados reais de uma transportadora de carga fracionada que atua em nível nacional. Essa aplicação possibilita que se tenha uma visão de quais os tempos de processamento esperados para uma aplicação real, além de ser um indicativo de qual a economia potencial em termos de custos operacionais que pode ser obtida com o uso dos algoritmos propostos.

5.1 Dados gerados aleatoriamente

Para a comparação entre os resultados do algoritmo proposto e o que seria um resultado ótimo obtido através de um modelo linear, foram geradas nove instâncias de pequeno porte.

As instâncias variaram de 5 a 10 nós, com custos e tempos aleatórios entre si. Também foram geradas demandas entre os nós da rede, com tamanhos e prazos também aleatórios.

A relação das instâncias em que foi aplicado o método pode ser vista na Tabela 10. Na Figura 38, por sua vez, são mostrados graficamente como são formados os grafos a partir dos dados.

Tabela 10: Relação de instâncias utilizadas

Instância	Nós	Arcos	Demandas
1	5	14	84
2	6	30	125
3	6	30	125
4	8	56	84
5	8	56	245
6	8	56	245
7	10	90	405
8	10	90	405
9	10	90	405

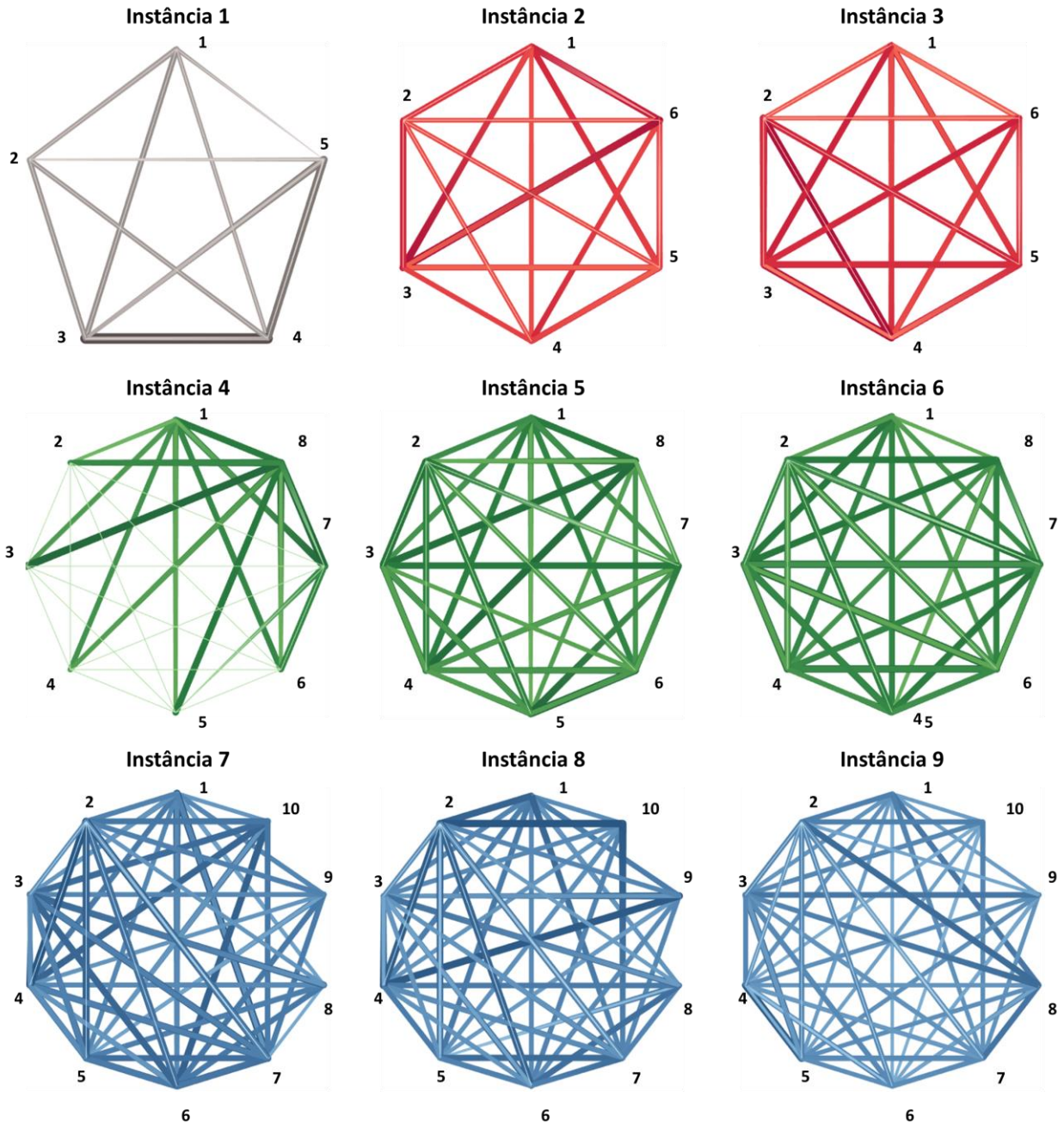

Figura 38: Redes das instâncias criadas

A cada instância, foi aplicado o algoritmo construtivo na sua versão sem restrições de unicidade de caminhos (Seção 4.2.1), com um total de 1.000 iterações. Para ordenação de demandas, foi utilizado um fator de 500% de aleatoriedade, que para os dados tem efeito prático de geração de ordenação totalmente aleatória. Após a heurística construtiva, foi aplicado o método de melhoria local descrito na Seção 4.3.1, com um total de 100 iterações.

Para cada instância, também foi aplicado o modelo linear descrito na Seção 3.1, sendo utilizado o pacote de otimização Gurobi®. Dada a complexidade do problema, para as instâncias 2 a 9, foi aplicado o GUROBI 7.0.2 em computadores de maior porte (2.27 GHz Intel(R) Xeon(R) E5520, 16 núcleos e 48 GB RAM). Para a instância 1, mais simples, o processamento foi feito em uma máquina com menor capacidade (2.20 GHz Intel Core i5 8GB RAM), que também foi utilizada no processamento da heurística. Para acelerar o processamento, a melhor solução encontrada nas heurísticas para cada instância foi utilizada como solução inicial para o programa.

A relação de resultados, tanto para o método heurístico quanto para a modelagem linear podem ser observados na Tabela 11. As colunas MIP e Heurística mostram os valores de função objetivo identificados para o modelo em programação inteira mista e o algoritmo proposto, respectivamente. Também são apresentados os tempos de processamento de cada um. Na coluna Gap é mostrado o valor de do *gap* da solução encontrada no MIP. Por fim, pode ser vista a diferença entre os valores de função objetivo entre as diferentes abordagens.

Tabela 11: Comparação de resultados entre heurística e modelos lineares

Instância	MIP	Gap	Tempo MIP (s)	Heurística	Tempo heurística (s)	Diferença
1	551	0%	-	551	2	0%
2	51.875	40%	43.200	51.875	3	0%
3	60.125	33%	43.200	60.500	3	1%
4	17.724	35%	43.200	17.724	3	0%
5	174.195	52%	43.064	177.135	3	2%
6	245.245	50%	43.165	246.470	3	0%
7	665.415	82%	38.645	665.415	7	0%
8	681.210	81%	39.893	684.855	7	1%
9	608.715	87%	36.882	608.715	7	0%

Como pode ser observado, mesmo com o processamento do modelo de programação linear inteira sendo realizado em máquinas de grande porte, os valores de *gaps* encontrados são altos. A única exceção foi a primeira instância, em que a solução ótima foi identificada. Isso se dá pelo fato de que essa instância possui uma rede significativamente menor que as demais, como pode ser observado na Figura 38. A

instância 4, com um número reduzido de demandas na rede frente às demais instâncias com 8 nós, também apresentou um *gap* inferior aos seus pares, conforme esperado, embora ainda seja acima de 30%.

Embora a distância teórica para os pontos ótimos seja grande, mesmo com horas de processamento, o modelo MIP foi capaz de produzir soluções com no máximo 2% menor que sua contrapartida heurística. Somente em dois dos casos (instâncias 5 e 8) o modelo linear foi capaz de encontrar uma solução melhor.

Dada a diferença significativa nos tempos de processamento entre modelo linear e heurística (o máximo tempo de processamento foi de 7 segundos, frente a um tempo mínimo do modelo linear de 10 horas), obtendo de maneira prática resultados similares, pode-se concluir que o algoritmo proposto se torna uma alternativa factível em termos de aplicação no mercado, ao contrário do método linear.

5.2 Rede real

Além dos dados fictícios, também foram coletados dados reais de uma transportadora de carga fracionada, com atuação nas regiões Sudeste e Sul do Brasil através de 33 terminais. A relação dos nós na rede pode ser observada na Tabela 12, enquanto a sua distribuição espacial é mostrada na Figura 39.

Foi recebido um histórico de um mês de demandas entre os diferentes terminais da rede, bem como todas as transferências de veículos realizadas no mesmo período, correspondendo à atual solução utilizada pela empresa nas suas operações.

Um primeiro ponto identificado nos dados é a heterogeneidade da frota empregada, diferindo do modelo proposto, em que todos os veículos são iguais em termos de capacidade. Foi necessária então uma normalização da solução atual, em que a quantidade de veículos heterogênea que percorre cada nó era transformada em uma quantidade estimada de veículos padrão. O fluxo de veículos normalizado entre as diferentes filiais pode ser observado na Figura 40, em que o espessura dos arcos é proporcional à quantidade de cargas transportada.

Tabela 12: Relação de filiais na rede real estudada

ID	Filial
1	Americana
2	Barra Mansa
3	Bauru
4	Blumenau
5	Campinas
6	Contagem
7	Divinópolis
8	Juiz De Fora
9	São Jose Do Rio Preto
10	São Paulo
11	Uberlândia
12	Curitiba
13	Três Lagoas
14	Viana
15	Cascavel
16	Caxias Do Sul
17	Criciúma
18	Londrina
19	Maringá
20	Nova Friburgo
21	Porto Alegre
22	Pouso Alegre
23	Presidente Prudente
24	Ribeirão Preto
25	Rio de Janeiro
26	Santa Cruz do Sul
27	Sao José dos Campos
28	Montes Claros
29	Tubarão
30	Governador Valadares
31	Teófilo Otoni
32	Praia Grande
33	Cajamar

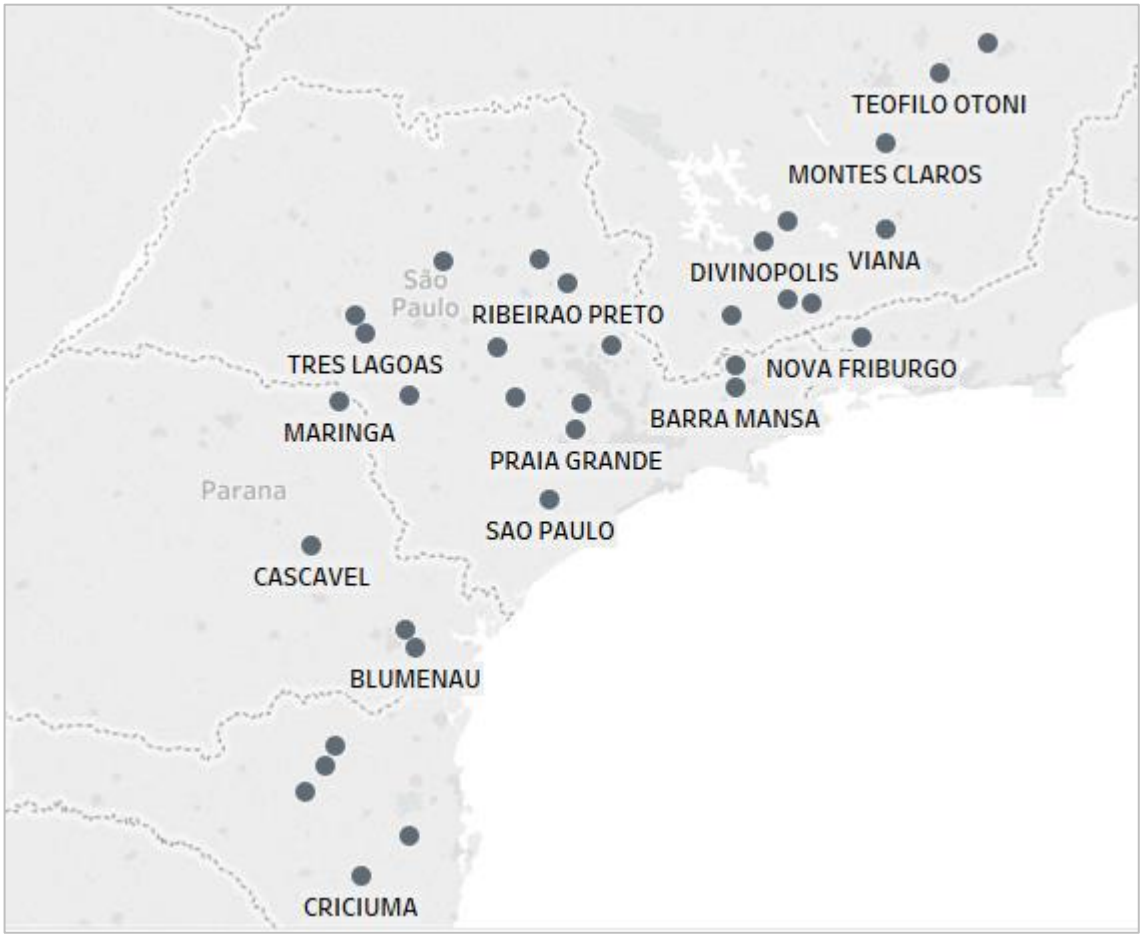

Figura 39: Filiais da rede real utilizada

Não foram disponibilizados os custos da operação, tanto de transporte quanto de movimentação dos terminais. Para estimar um valor aproximado, foram utilizados como referência os custos médios de transporte disponibilizados em Hellmuth (2004), apresentados na Tabela 13. No trabalho, é realizado um estudo de custos fixos e variáveis de frota, variando a frequência com que uma determinada transferência é feita. Por exemplo, arcos na rede que tenham uma frequência de envio menor do que 12 embarques no mês são classificadas como baixa frequência, tendo um custo por km rodado de 1,1846.

Embora esses dados não sejam atualizados ou calculados para a transportadora em questão, não é propósito do presente trabalho a estimativa precisa dos custos operacionais atuais presentes, e sim gerar um número que represente de maneira

consistente o nível de custos atuais e que permita a comparação com outros cenários de operação.

Figura 40: Fluxos empregados na solução atual da empresa

Tabela 13: Custos de transporte por tipo de linha

Tipo de linha	Frequência mensal	Custo/km
Baixa	Até 12 transferências	1,1846
Média	Entre 12 e 20 transferência	1,3868
Alta	Maior do que 20 transferências	1,3844

No caso dos custos de movimentação, foi utilizado um valor médio semelhante ao observado no trabalho de Hellmuth (2004). Para diferenciar o custo entre os terminais, foi realizado um ajuste de acordo com sua volumetria, fazendo com que terminais com maior volume de cargas tivessem custos unitários abaixo da média, uma vez que, em teoria, teriam maior porte e, consequentemente, infraestruturas mais eficientes.

Ao final do processo, foi estabelecido um custo de referência global de 652.786, dividido entre suas parcelas de transporte e movimentação conforme o mostrado na Tabela 14.

Tabela 14: Cenário de custos atual do caso real

Parcela	Valor
Custo transporte	529.823
Custo movimentação	122.963
Custo total	652.786

Uma vez realizados os ajustes necessários nos dados, foi possível aplicar os algoritmos descritos no Capítulo 4. Para tanto, foram aplicados diferentes cenários de ordenação de demandas para geração de cenários aleatórios, com fatores de aleatoriedade de 15%, 25%, 50%, 100% e 500%.

Para cada cenário, foi implantada uma heurística construtiva com 1.000 iterações, ou seja, 1.000 casos com ordenação de demanda aleatória de acordo com o fator de aleatoriedade previsto. Após a fase de construção, era processada uma fase de melhoria local, com 100 iterações subsequentes.

Os resultados finais de custos para cada uma das abordagens (sem restrições, caminhos fixos e caminhos por dia da semana) e cada um dos fatores de aleatoriedade podem ser observados na Tabela 15. Nela, para cada tipo de abordagem (caminhos fixos, por dia da semana, ou sem restrições) e cada fator de aleatoriedade, são apresentados os valores de função objetivo ao final da fase construtiva do algoritmo e da melhoria local. Também é apresentada a redução de custo frente ao cenário atual de custos (652.786).

Tabela 15: Resultados obtidos para os diferentes processamentos

Abordagem	Aleatoriedade	Construtivo	Melhoria local	Redução cenário atual
Caminhos fixos	15%	498.883	423.513	35%
Caminhos fixos	25%	503.318	439.202	33%
Caminhos fixos	50%	507.138	430.064	34%
Caminhos fixos	100%	480.088	421.941	35%
Caminhos fixos	500%	486.578	403.029	38%
Dia da semana	15%	465.704	404.612	38%

Abordagem	Aleatoriedade	Construtivo	Melhoria local	Redução cenário atual
Dia da semana	25%	478.070	415.491	36%
Dia da semana	50%	465.681	412.290	37%
Dia da semana	100%	467.587	412.320	37%
Dia da semana	500%	468.731	403.990	38%
Sem restrições	15%	348.903	309.017	53%
Sem restrições	25%	348.624	310.692	52%
Sem restrições	50%	348.304	307.678	53%
Sem restrições	100%	343.971	311.359	52%
Sem restrições	500%	347.122	310.387	52%

Como pode ser visto, as três abordagens produzem reduções significativas frente ao cenário atualmente praticado pela empresa. Em especial, o cenário sem restrições em relação à unicidade de caminhos é capaz de produzir uma redução potencial de 52% frente aos custos estimados atualmente. Esse tipo de redução indica que o esforço em aplicação de rotas não fixas pode ser benéfico para companhias em geral, embora essa seja uma operação significativamente mais complexa de se planejar, gerenciar e controlar.

Já os resultados de cenários com rotas fixas produzem reduções previstas em torno de 35%. Essa proposta se assemelha à operacionalização de transportes atualmente praticada por transportadoras de carga fracionada, sendo portanto a sua implantação mais simples.

Já a terceira abordagem proposta, em que os caminhos percorridos são fixos de acordo com o dia da semana, apresentou resultados similares aos casos de caminhos fixos, com uma leve redução nos custos totais. Esse resultado diverge do descrito em Erera *et al.* (2012), em que essa variação do modelo apresentava reduções 60% maiores que o modelo de caminhos fixos.

Isso pode ser um efeito da forma com que o algoritmo de caminhos por dia da semana foi construído. Como a heurística construtiva retira demandas de rotas quando há qualquer mudança nas chegadas de rotas em termos de dias de semana, há uma redução nas movimentações consideradas como factíveis, limitando também o potencial de ganho.

Além de custos, um dos pontos interessantes para a comparação entre as diferentes abordagens é a quantidade de arcos da rede utilizada na solução, como pode ser observado nas Figuras Figura 41, Figura 42 e Figura 43, em que são apresentadas as rotas propostas (em laranja, verde e vermelho) e o que é atualmente praticado (em azul). Já a quantidade de arcos utilizada para cada um pode ser vista na Tabela 16.

Tabela 16: Quantidade de arcos utilizados para cada solução final encontrada

Caso	Arcos
Original	134
Sem restrições	242
Caminhos fixos	134
Dia da semana	234

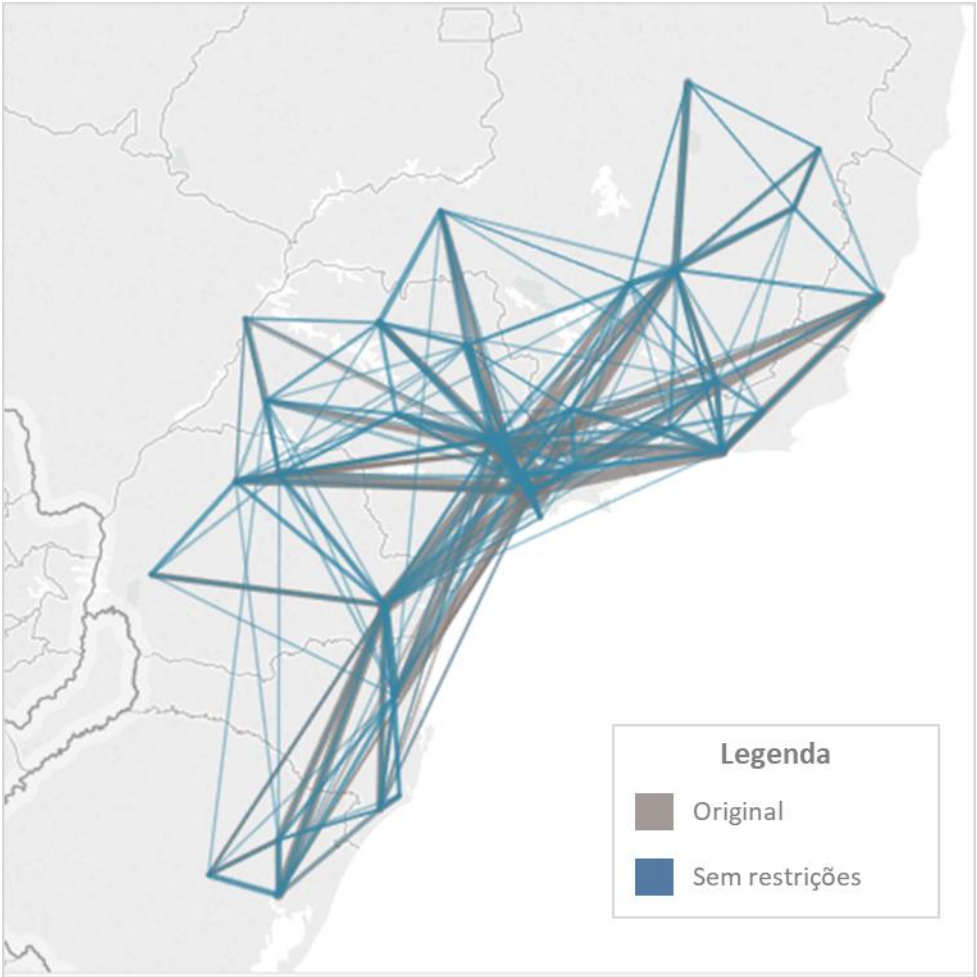

Figura 41: Comparação entre solução atual e modelo sem restrições de unicidade de caminhos

Como esperado, a variação do problema que não restringe caminhos faz maior uso de arcos, com um total de 242, frente aos 134 utilizados atualmente pela empresa. A sobreposição entre as duas soluções permite visualizar como diferentes arcos entre terminais secundários (Blumenau e Santa Cruz do Sul, Maringá e Cascavel) são aproveitados sem a restrição.

Como mencionado anteriormente, o sistema proposto pelo algoritmo sem restrições é significativamente mais complexo de se operacionalizar. Operando com sistemas fixos de caminhos, todas as mercadorias que compartilham o mesmo destino irão ser expedidas para o mesmo local, facilitando o processo de movimentação e carregamento de veículos nas docas.

Na solução encontrada, a média de diferentes caminhos seguidos por mercadorias que compartilham destino nos terminais na rede é de 2,3. Isso faz com que os operadores dentro dos terminais tenham que ter um controle maior de quais são as mercadorias tratadas e um processo de expedição flexível e preciso, o que deve aumentar os custos de movimentação nos terminais. Dado o ganho potencial observado com esse tipo de operação e o crescente uso de tecnologias como RFID em gerenciamento de mercadorias em centros de distribuição e seu consequente barateamento (conforme citado por Erera (2012)), essa ainda pode ser uma mudança bastante benéfica para os operadores.

Em relação ao cenário que exige caminhos fixos, o uso de arcos é igual à solução atual (134 arcos). O ganho encontrado se dá quase que exclusivamente com a revisão das linhas e a consequente seleção de arcos com maior possibilidade de sinergia entre demandas e menor custo. A sobreposição das duas soluções é capaz de exibir como são reduzidos arcos com grandes volumes e utilizados arcos secundários de maneira mais extensiva (caso de Ribeirão Preto e Londrina).

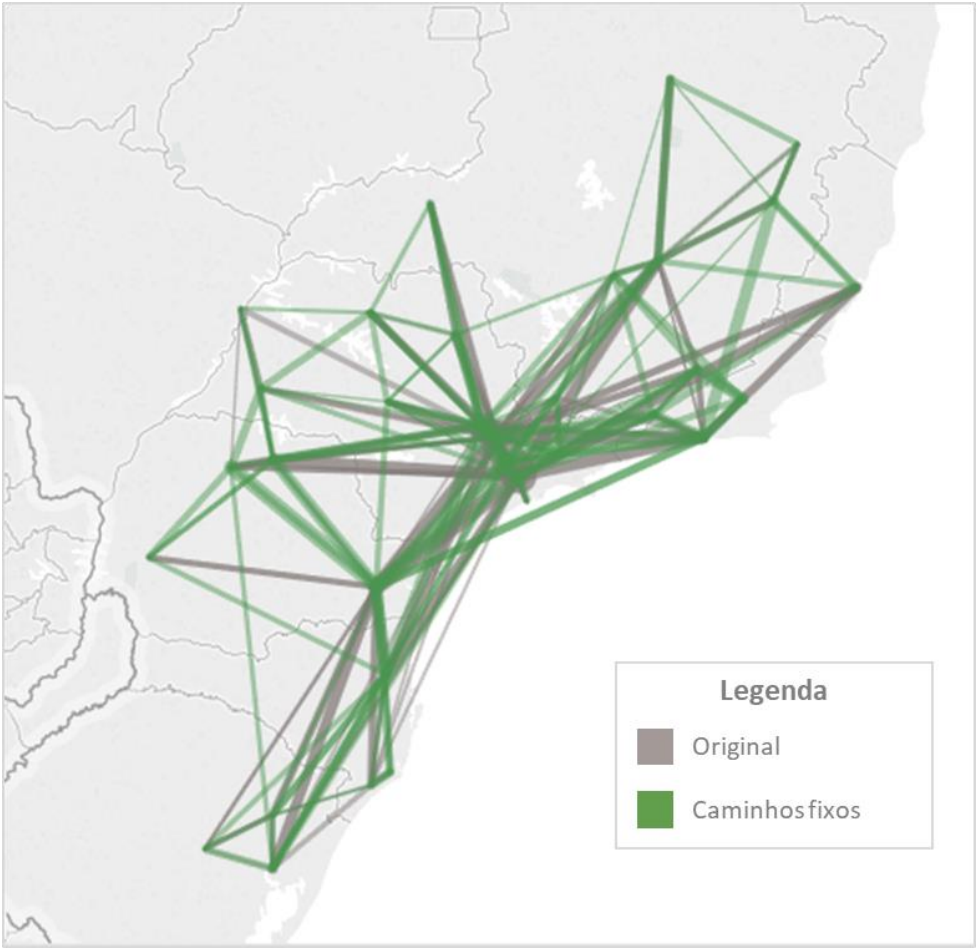

Figura 42: Comparação entre solução atual e modelo com caminhos fixos

Dado que são geradas rotas fixas semelhantes ao que hoje é praticado por transportadoras no mercado, a implantação desse tipo de solução é mais simples do que o caso anterior.

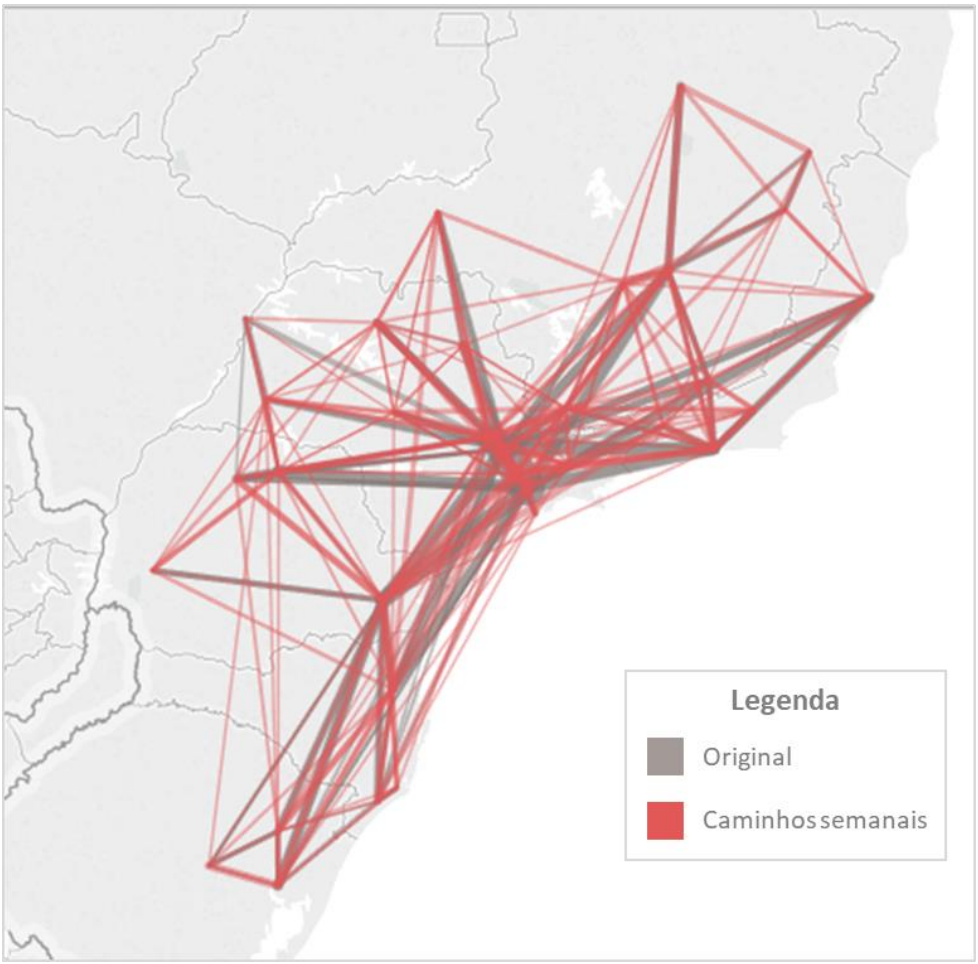

Figura 43: Comparação entre solução atual e modelo com caminhos por dia da semana

Já o terceiro caso se mostra como um intermediário entre os dois anteriores. O seu uso de arcos é abaixo do cenário totalmente livre, porém muito acima dos arcos fixos. A sobreposição em relação ao praticado atualmente demonstra como o uso de arcos é bastante superior.

Em termos de operacionalização, essa solução não necessita de um nível de controle dentro dos terminais como o modelo sem restrições de caminho, uma vez que não precisa realizar o rastreamento exato de cada uma das encomendas. A sua implantação se daria pela mudança no planejamento de expedição e posicionamento de docas de acordo com o dia da semana. Dessa maneira, as alterações necessárias para a aplicação de um sistema desse gênero não envolveriam necessariamente grandes mudanças sistêmicas ou tecnológicas, e sim à capacitação das equipes e armazéns para suportar mudanças regulares no seu planejamento.

Outros dois fatores foram analisados em termos de resultados, além dos custos finais. Um deles é a quantidade de iterações utilizadas no algoritmo, identificando qual seria uma quantidade ideal para ser estabelecida como parâmetro de parada, garantindo um equilíbrio em termos de performance e redução de custos.

Outro ponto avaliado é o fator de aleatoriedade utilizado para a geração de casos. É esperado que, fatores com menor valor devem direcionar resultados melhores mais rapidamente, mas podem não ser capazes de apresentar um melhor resultado global. Por outro lado, o aumento da aleatoriedade deve ser capaz de produzir melhores resultados, embora precise de mais iterações para chegar a resultados adequados.

Ao analisar os resultados para os cenários sem restrições de unicidade de caminhos (Figura 44), é possível observar que os valores de aleatoriedade pouco afetam o resultado final obtido na heurística construtiva. A máxima diferença de custo encontrada é de pouco mais de 1%. Embora os dois casos com menor custo sejam os de aleatoriedade de 100% e 500%, respectivamente, é uma diferença pouco significativa frente aos demais.

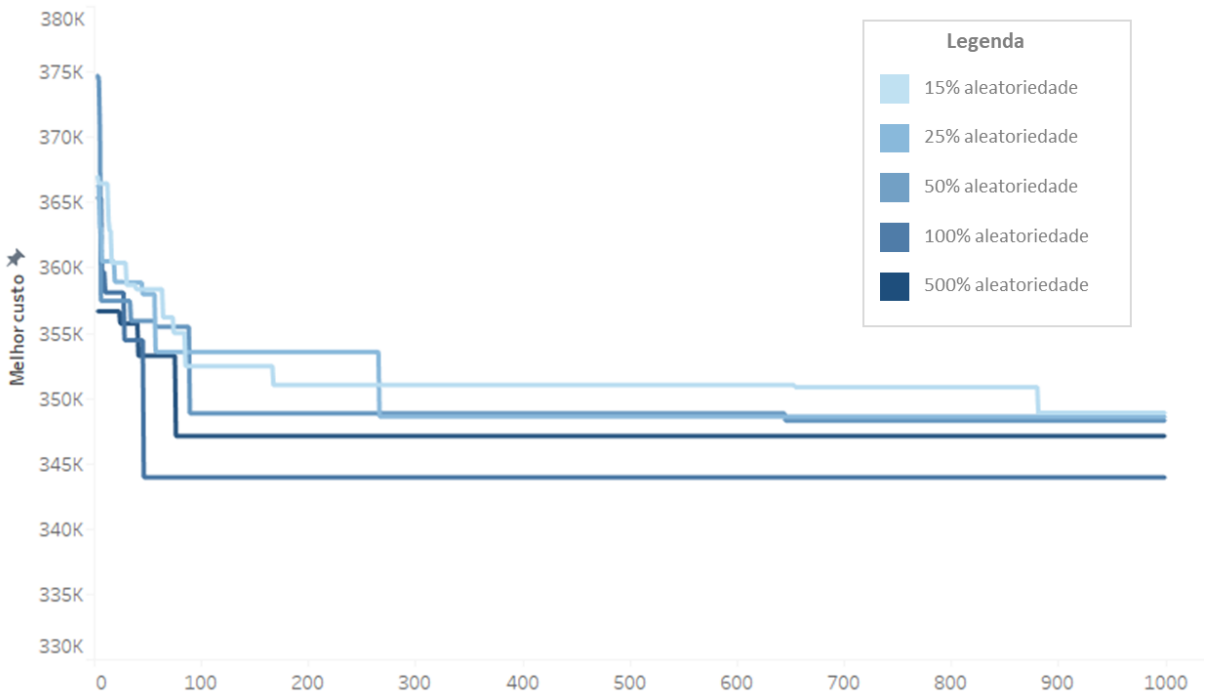

Figura 44: Resultados do cenário sem restrições de caminho

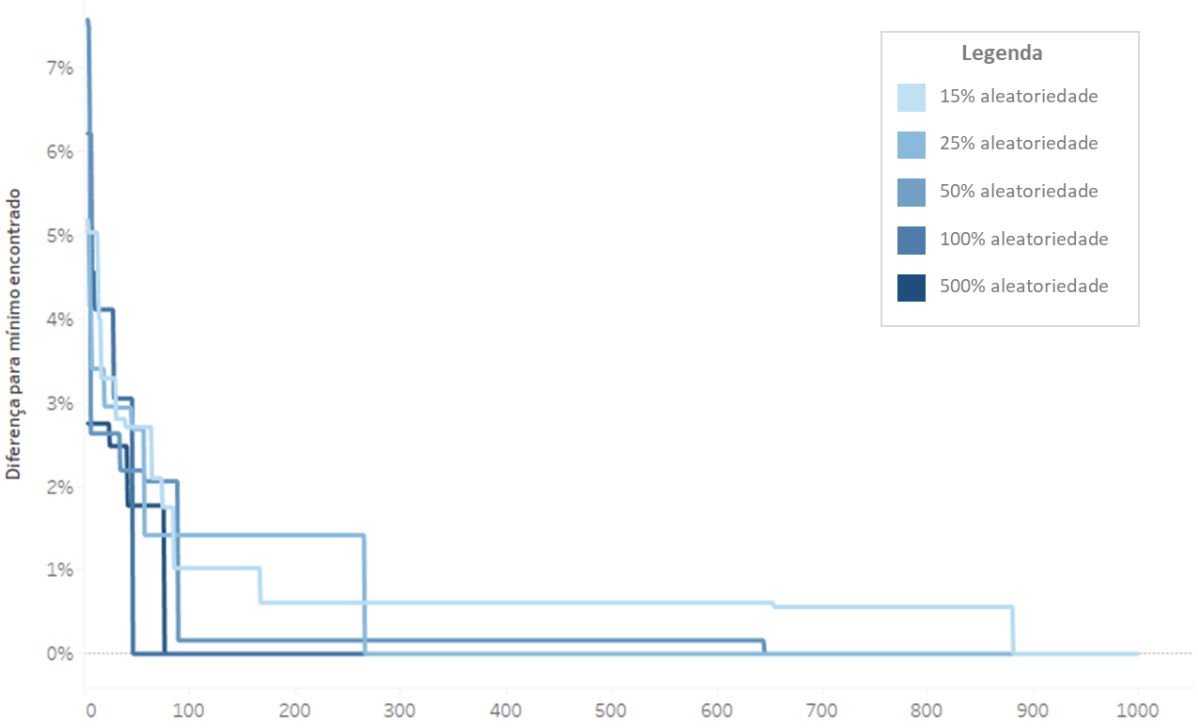

Figura 45: Variação frente ao mínimo encontrado para cenário sem restrições

Isso sugere que, para o caso sem restrições na unicidade de caminhos, o valor de aleatoriedade pouco influi no resultado final. Isso pode ser explicado pelo fato de, como as demandas são livres, a aleatoriedade deverá produzir somente diferenças nas sinergias entre as demandas que o modelo será capaz de captar.

Para todos os cenários, as diferenças entre os primeiros valores obtidos no modelo e o final encontrado na construção das soluções é de 4%, reforçando o ponto anterior. Após 100 iterações, essa diferença cai um máximo de 1%, como visto na Figura 45. A partir de 800 iterações, quase não há qualquer ganho em termos de custos para todos os cenários.

Para os casos caminhos fixos, porém, o panorama é diferente. Como pode ser observado na Figura 46, os cenários com maior grau de aleatoriedade produzem resultados melhores ao final do processo, com uma diferença entre o pior e o melhor caso chegando a 6%. Após 200 iterações, as diferenças entre a solução vigente de cada cenário e o seu valor final é de 2% (Figura 47). A partir de 300 iterações, não são produzidas melhorias nas soluções.

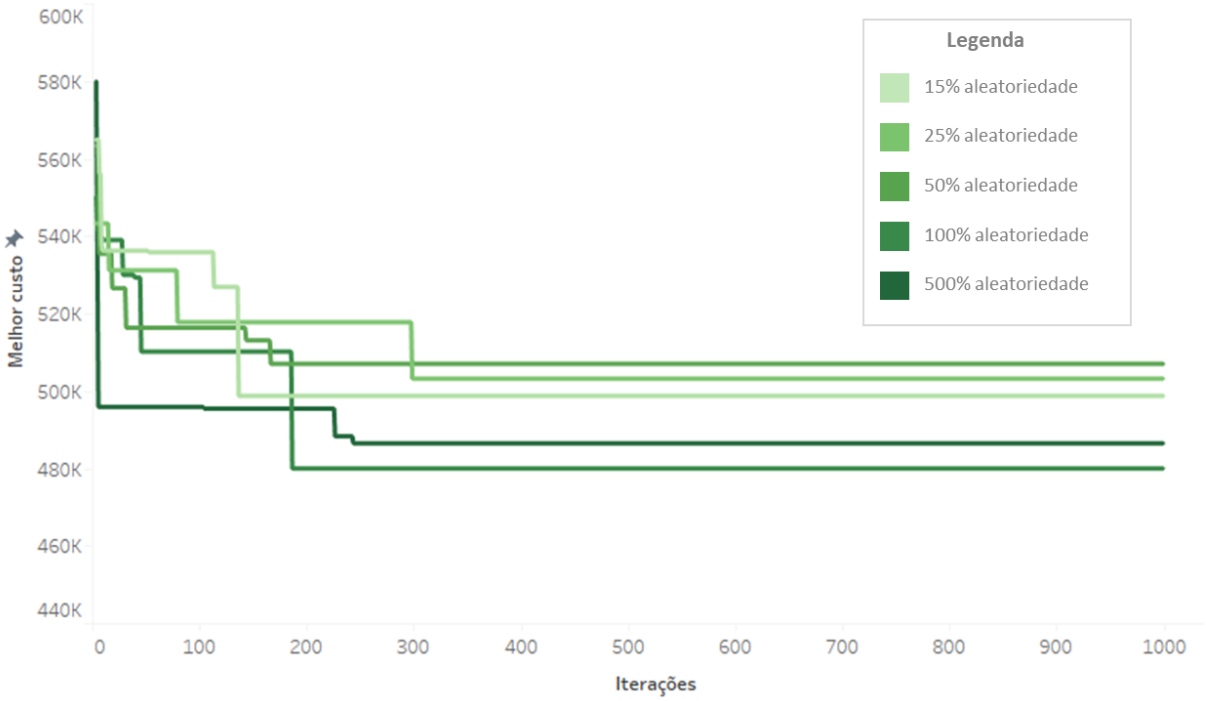

Figura 46: Resultados do cenário com caminhos fixos

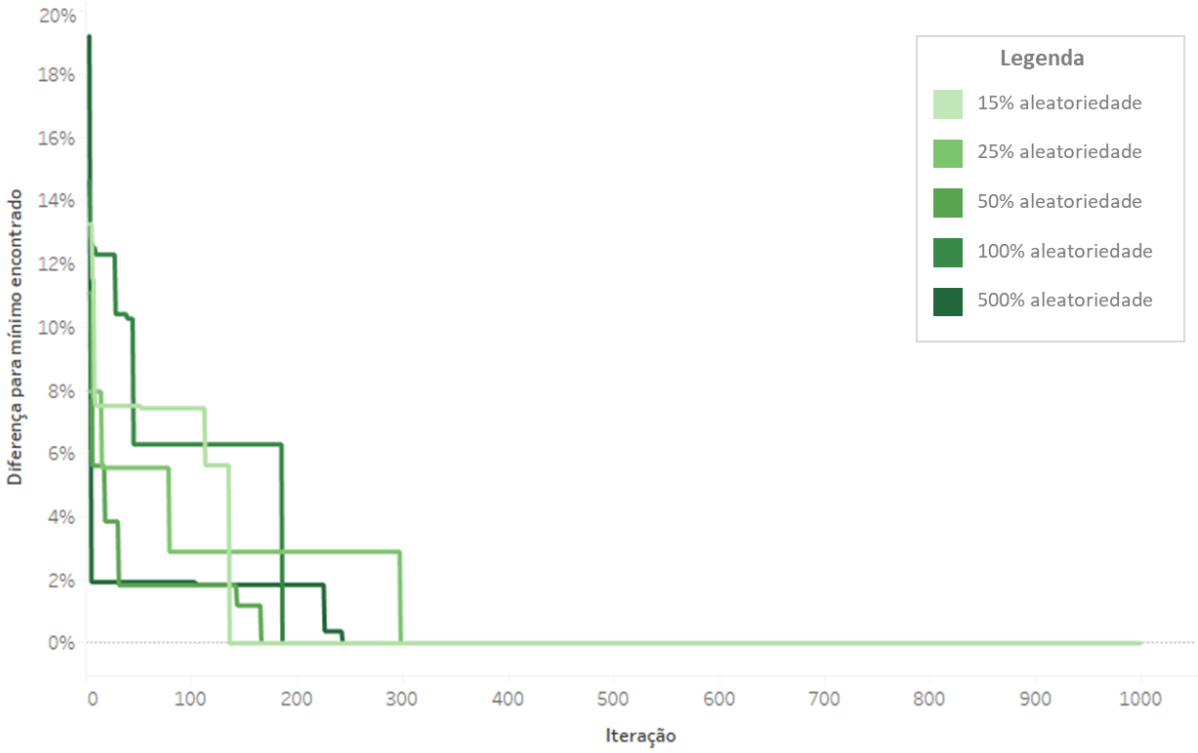

Figura 47: Variação frente ao mínimo encontrado para cenário com caminhos fixos

Esse resultado pode ser explicado pelo fato de, no caso de caminhos fixos, a ordem com que as demandas são processadas no algoritmo influencia de maneira significativa o resultado final do modelo, uma vez que as primeiras demandas irão determinar os caminhos fixos que as demais deverão percorrer. As últimas demandas a serem consideradas no sistema, na sua maioria, terão todos os caminhos pré-definidos já estabelecidos, de maneira que a sua margem para economias se torna bastante limitada. Dessa maneira, diferentemente do primeiro caso, em que a aleatoriedade influenciava a sinergia com que as demandas se combinavam, nesse caso há a forte influência de quais são as demandas dominantes em termos de construção de caminhos. É de se esperar, portanto, que quanto mais variados os casos, maior será a redução de custos encontrada, explicando o porquê dos casos com aleatoriedade maior tenham apresentado melhores resultados.

No caso de caminhos por dia da semana, foi identificado um comportamento semelhante ao da abordagem sem restrições. As diferenças entre o melhor e o pior cenário de aleatoriedade foi em torno de 2,5% (Figura 48). Os melhores cenários foram os com aleatoriedade de 15% e 50%, demonstrando que o aumento na aleatoriedade nesses casos não produziu melhoria nos custos identificados. Excluindo o caso de 25%, a diferença entre os demais é de menos de 1%, o que reforça que a aleatoriedade não é um fator determinante no valor final de custo.

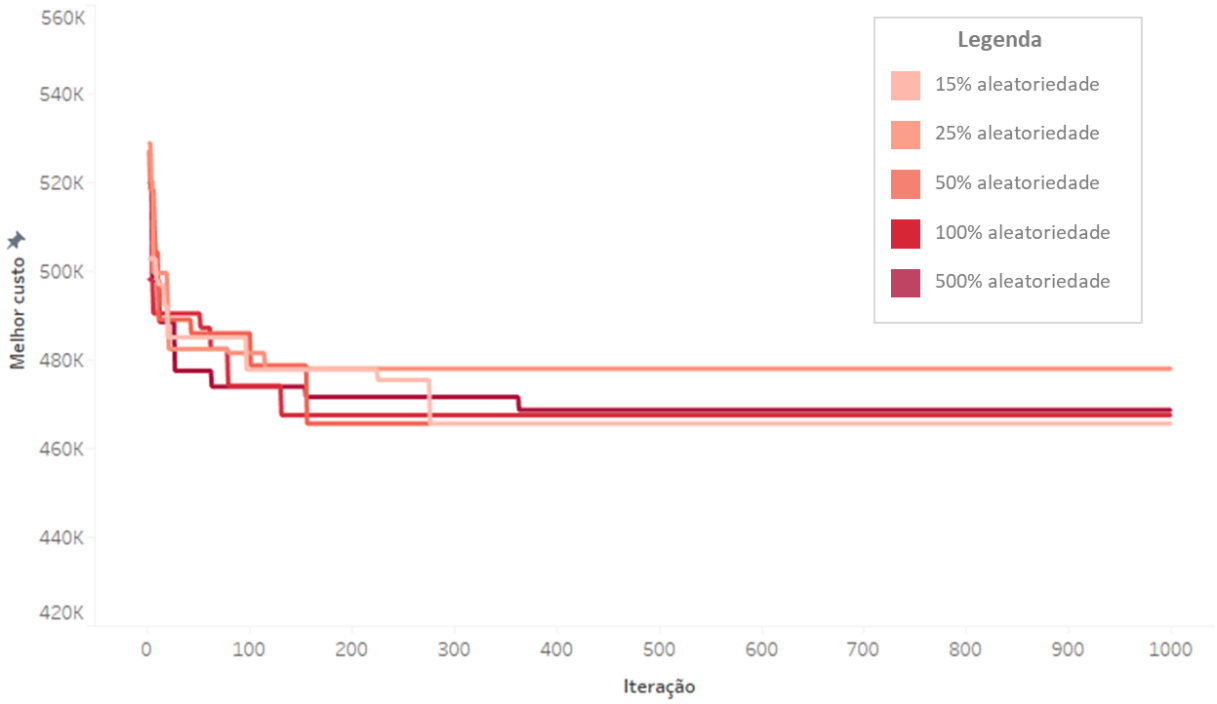

Figura 48: Resultados do cenário com caminhos por dia da semana

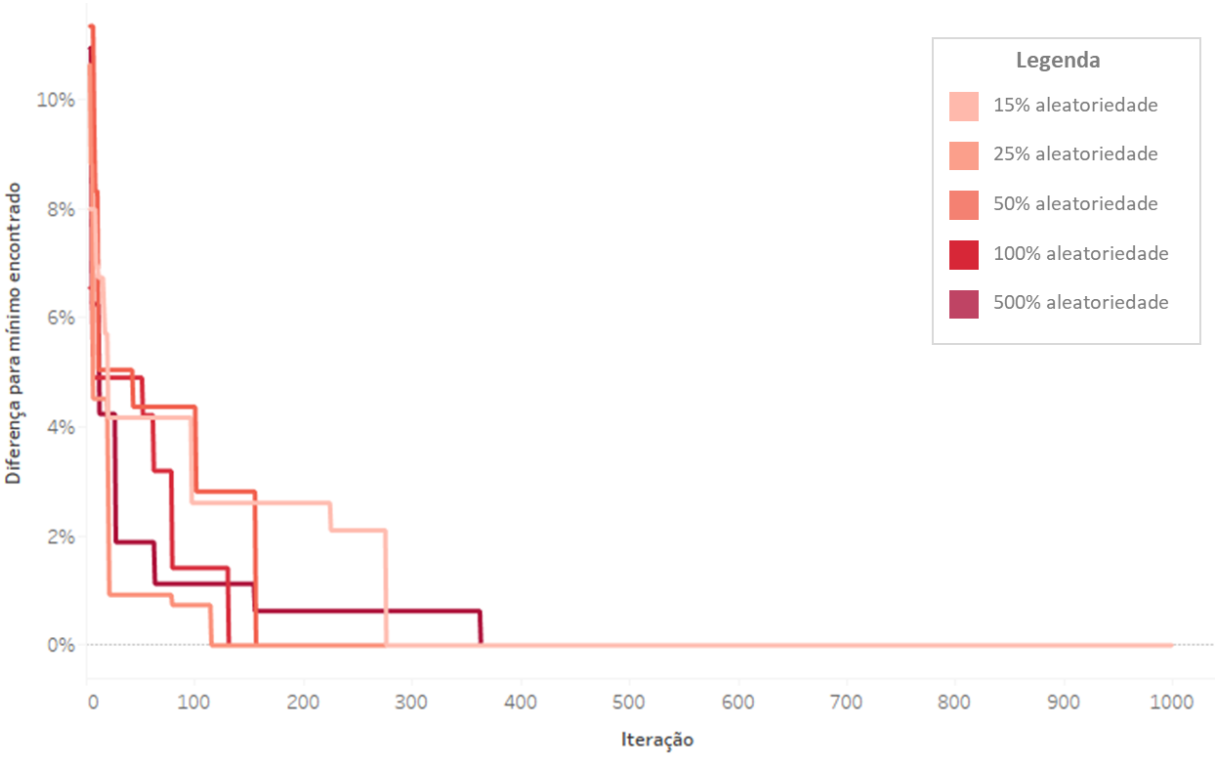

Figura 49: Variação frente ao mínimo encontrado para cenário com caminhos por dia da semana

A maior parte dos cenários se estabilizou com 200 iterações, com nenhuma melhoria identificada a partir da iteração 400, conforme visto na Figura 50.

O caso de caminhos por dia da semana, portanto, embora também gere caminhos pré-determinados, não apresentou grande influência em relação à ordenação das demandas. Isso pode ser explicado pela maior flexibilidade nesse cenário, dado que o caminho só será definido para uma demanda caso ela tenha o mesmo dia da semana na chegada que uma outra demanda anterior. Dessa maneira, a influência de qual a ordem de processamento das demandas diminui.

Após as iterações iniciais referentes ao algoritmo construtivo, se iniciam as buscas locais. As três abordagens apresentaram reduções significativas nas funções objetivo nas primeiras iterações, chegando a uma convergência rapidamente.

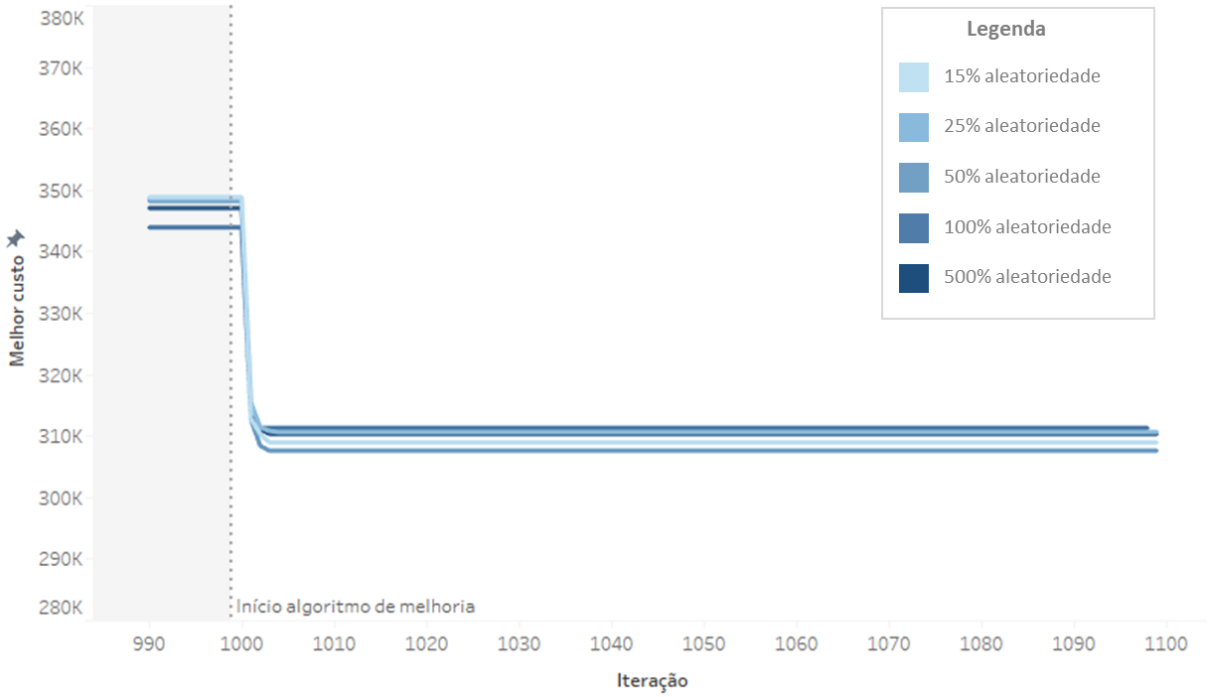

Figura 50: Função objetivo com melhorias locais para caso sem restrições

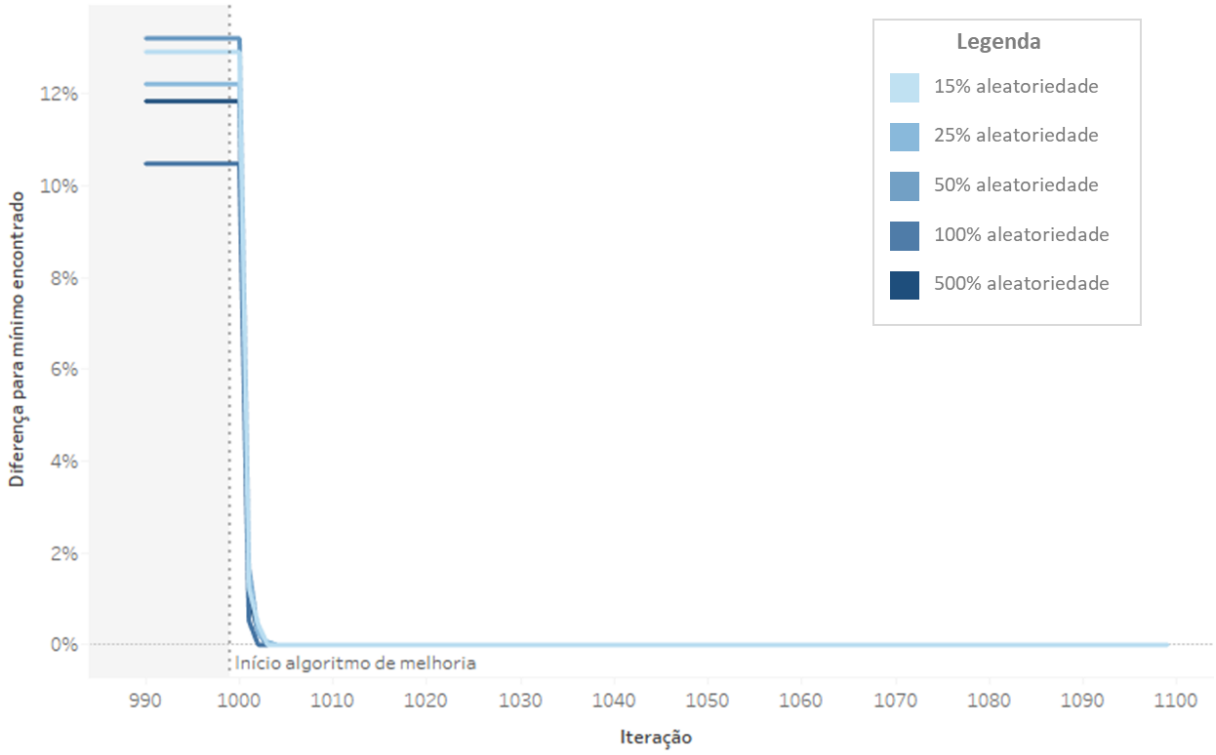

Figura 51: Variação da função objetivo para melhorias locais no caso sem restrições

No caso sem restrições de unicidade, como pode ser observado na Figura 50 e Figura 51, a busca local foi capaz de reduzir o valor da função objetivo entre 10% e 12% frente ao mínimo obtido durante o algoritmo construtivo. Novamente, as soluções finais ficaram muito próximas entre si na comparação entre valores de aleatoriedade. Enquanto o cenário com fator de aleatoriedade de 100% apresentou um custo final de 311.359 (maior encontrado), o cenário com menor valor, com fator de 100%, apresentou um custo de 307.678, mostrando uma diferença de 1%.

Além disso, em todos os cenários foi possível observar uma convergência rápida em termos de iterações. Mesmo processando um total de 100 iterações, em todos os casos o melhor resultado já tinha sido encontrado em 5 iterações. Isso demonstra que a busca local é capaz de produzir reduções nos custos totais, embora a sua influência seja limitada.

Para o algoritmo com restrições de caminhos fixos, o que se observa é uma redução bastante significativa na função objetivo com a heurística de melhoria local, chegando a 20% de redução frente ao valor obtido durante a heurística construtiva (valores

absolutos na Figura 52 e relativos na Figura 53). Isso pode ser explicado pela influência que a ordenação das demandas tem sobre a solução, e ao remover demandas e seus respectivos caminhos fixos para reinseri-las, é possível ao algoritmo "corrigir" decisões erradas em termos de estabelecer caminhos fixos. Esse fator não é tão significativo para o caso sem restrições de unicidade, uma vez que cada demanda tem a "liberdade" de seguir qualquer caminho na rede, de maneira similar ao já discutido anteriormente em referência às diferenças entre os cenários de aleatoriedade.

Novamente, porém, é observado que o algoritmo chega a uma solução convergente logo após algumas iterações. De maneira similar ao modelo anterior, após 5 iterações, já é obtida a solução mínima em termos de custo.

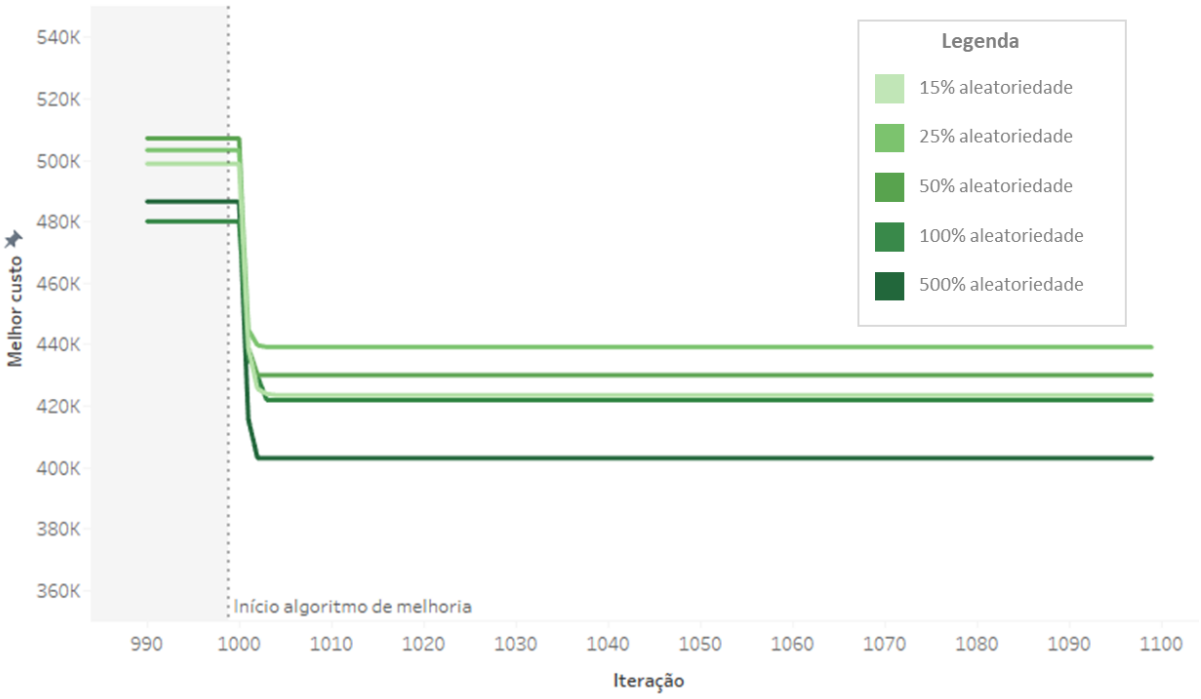

Figura 52: Função objetivo com melhorias locais para caso de caminhos fixos

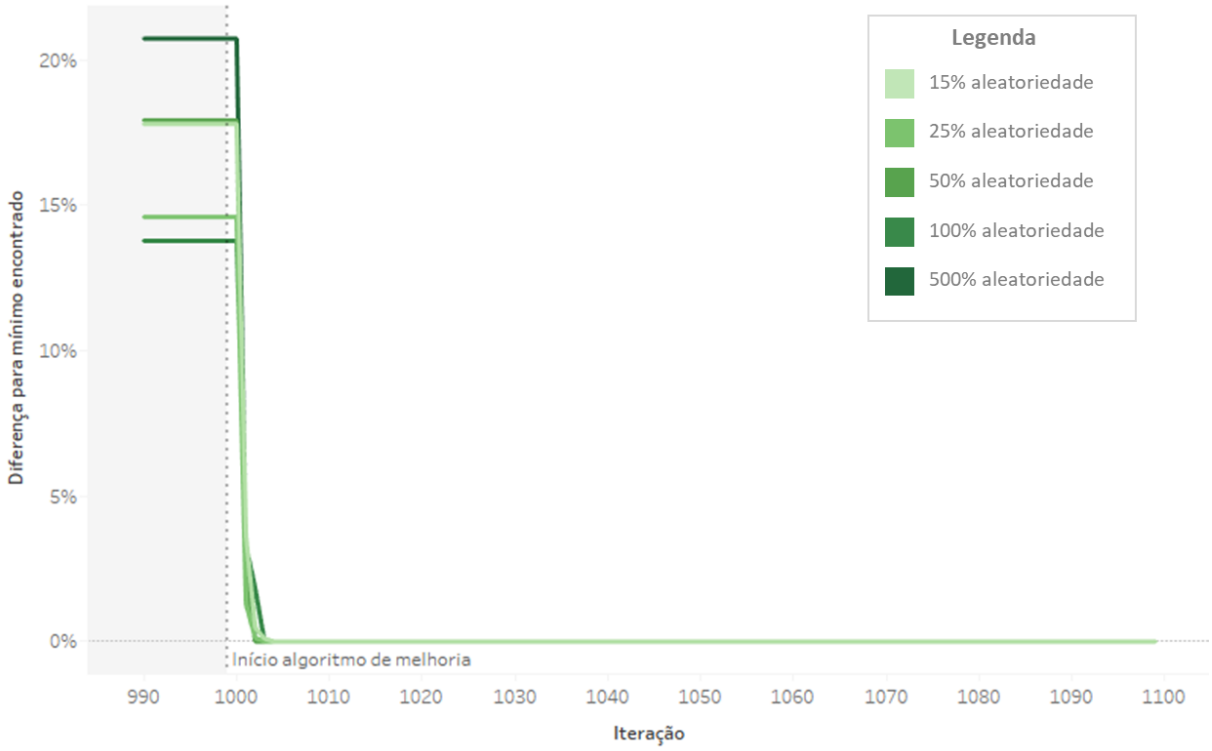

Figura 53: Variação da função objetivo para melhorias locais no caso de caminhos fixos

Finalmente, nas soluções com restrições semanais, é encontrado um comportamento intermediário, com reduções no valor de função objetivo chegando a 16% (Figura 54 e Figura 55). Embora um comportamento intermediário seja esperado, é possível concluir que a melhoria local proposta para esse tipo de modelo (Seção 4.3.3) não apresenta a mesma efetividade do caso de caminhos fixos.

Isso se dá pelo fato do algoritmo com caminhos por dia da semana ter apresentado um resultado praticamente equivalente ao de caminhos fixos, que é mais restrito. Ao observar o resultado durante a parte construtiva, essa diferença é em média de 5%. A menor eficácia na melhoria local para caminhos semanais, portanto, faz com que as soluções finais sejam muito semelhantes.

Essa falta de eficiência na busca local pode ser explicada novamente pela complexidade no algoritmo proposto, ligado ao processo de retirada e reinserção de demandas, que limita o espaço de busca de soluções factível.

De forma similar aos demais cenários, também foi observada uma convergência de resultado após 5 iterações. Pode-se concluir que um máximo de 10 iterações de

processamento de melhorias locais deve ser suficiente para todos os modelos identificados.

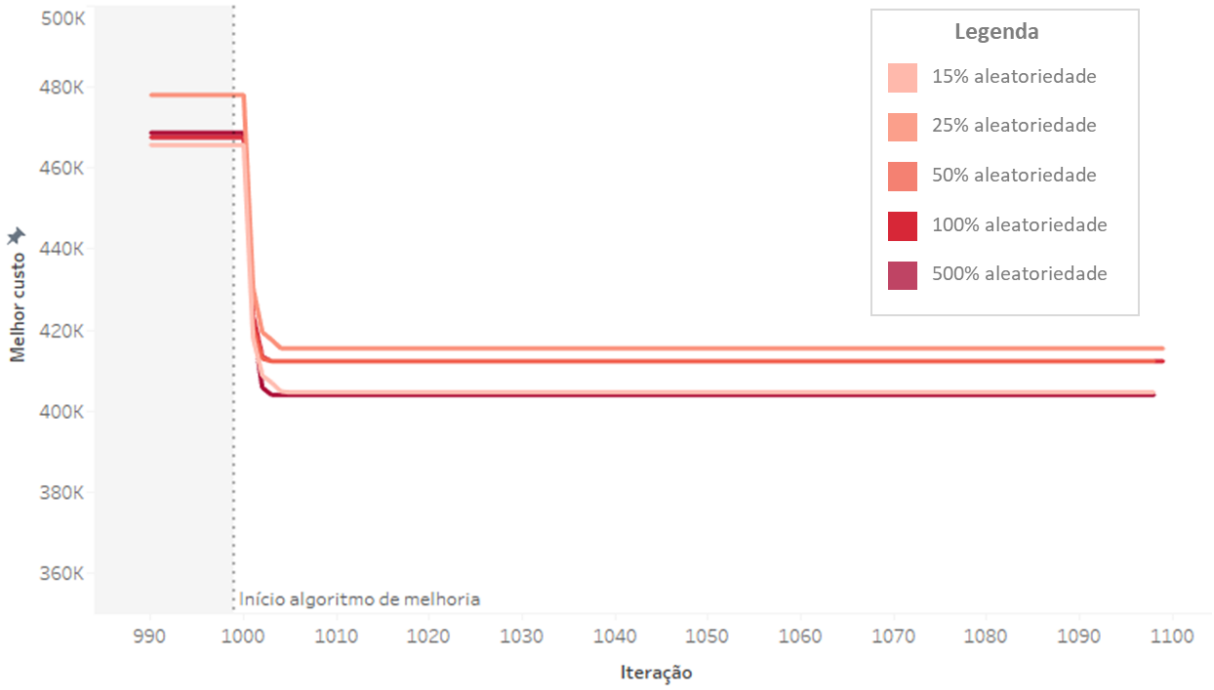

Figura 54: Função objetivo com melhorias locais para caso de caminhos por dia da semana

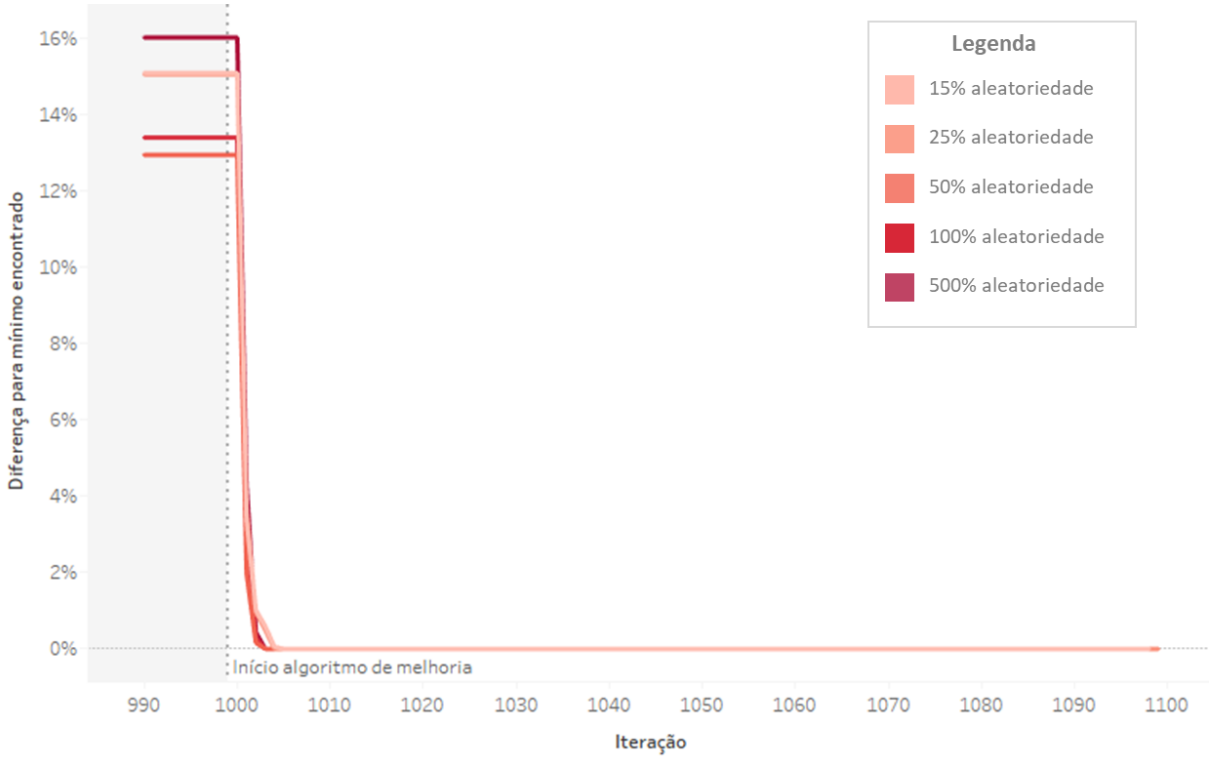

Figura 55: Variação da função objetivo para melhorias locais no caso sem restrições

5.3 Discussões

Com a aplicação do algoritmo, sejam dados gerados aleatoriamente ou reais, é possível realizar algumas comparações com outros trabalhos existentes na literatura, em especial em relação à sua aplicabilidade e potencial de redução de custos.

Um primeiro ponto a se destacar é que tanto a modelagem quanto as heurísticas propostas tratam a restrição temporal de maneira contínua, ou seja, sem uma discretização do tempo.

Diferentes trabalhos anteriores, como Jarrah *et al.* (2009) e Zhang (2010), basearam seus modelos em grafos que representavam não só a distribuição espacial dos nós na rede, mas também a sua localização temporal. Para a geração desses grafos, é necessária uma discretização temporal, em que o seu horizonte de programação é dividido em diferentes pontos inteiros.

Ao realizar a discretização, inicia-se um debate sobre qual seria o seu valor ideal. No trabalho de Jarrah *et al.* (2009), cada ponto temporal representa um dia. No trabalho

de Erera *et al.* (2012), esse valor é questionado, e então proposta uma discretização de 8 períodos ao dia. Por fim, Zhang (2010) utiliza uma discretização horária, ou seja, 24 períodos ao longo do dia.

A discussão sobre qual deve ser o nível de detalhe na discretização temporal pode ser bastante ampla, uma vez que diferentes operações irão necessitar, a rigor, de diferentes discretizações. No limite, poderíamos obter uma divisão horária ou mesmo de minuto a minuto, garantindo que atenderia a qualquer gênero de transportadora e operação. Isso, porém, faz com que o grafo de análise cresça na mesma medida, o que poderá afetar significativamente os tempos de processamento e a posterior análise.

Os modelos lineares e métodos propostos nos capítulos 3 e 4, em contrapartida, tratam o tempo de maneira contínua, o que elimina discussões a respeito do nível de discretização que deve ser utilizado. No presente trabalho, cada demanda pode ser alocada a qualquer horário de partida ou chegada, podendo chegar ao nível de minutos, embora as operações dificilmente cheguem nesse detalhe.

Vale ressaltar que esse tratamento contínuo não teve como consequência um aumento no tempo computacional de processamento. Como deverá ser discutido posteriormente, os tempos de processamento observados são semelhantes aos reportados em outros trabalhos.

Em relação à modelagem matemática linear, como esperado, foi encontrada uma grande dificuldade em resolver o problema de maneira ótima, reforçando a visão de que se trata de um problema com complexidade NP-Difícil (Jarrah (2009)). Dada a impossibilidade de se processar uma instância real, foram propostas pequenas instâncias que permitiriam uma comparação dos resultados e, para acelerar os processamentos, a melhor solução encontrada pela heurística foi fornecida como solução inicial.

Das nove instâncias propostas, apenas uma chegou ao ponto ótimo após um dia de processamento. Para esse caso, o resultado é equivalente ao obtido pela heurística.

Já para as demais instâncias, foram encontrados *gaps* elevados, chegando a 80% para as instâncias com maiores nós. Apenas dois casos conseguiram identificar uma solução melhor do que a inicial, com redução de no máximo 2% no valor de função objetivo.

Esses resultados reforçam, primeiramente, a complexidade do problema. Mesmo pequenas instâncias não foram capazes de chegar a pontos ótimos após 1 dia de processamento em máquinas de alta performance.

Em relação à comparação entre a heurística e o resultado ótimo, não é possível afirmar qual a distância esperada para o ótimo, dado o grande valor de *gaps*. Porém, poucos casos conseguiram identificar uma solução melhor do que a inicial, o que sugere que o resultado encontrado pela heurística seja relativamente bom em termos de função objetivo.

Ao estender a aplicação dos algoritmos para um caso de transportadora brasileira, é possível obter uma visão a respeito da aplicabilidade dos modelos propostos a ambientes reais, tanto do ponto de vista dos tempos de processamento e adaptação de dados quanto dos ganhos esperados.

Conforme visto na Tabela 17, para as três variações de heurísticas propostas, o ganho potencial frente ao custo atual da transportadora é bastante significativo. Com valores de 38%, é possível afirmar que a implantação de um sistema do gênero, ainda que não consiga capturar todos os ganhos potenciais previstos na solução ideal matemática, deve produzir resultados significativos em termos de redução de custos para a empresa.

Em comparação com demais trabalhos realizados, a redução prevista é maior do que o geralmente reportado. Isso pode ser explicado pelo fato de que a maior parte das empresas citadas nos demais trabalhos já possuíam um sistema próprio de desenho de rotas, fazendo com que a redução de custos seja marginal. Muitas empresas brasileiras não possuem nas suas operações sistemas já implantados do gênero, que faz com que o seu benefício potencial seja maior, já visto no trabalho de Hellmuth (2004).

Tabela 17: Comparação de resultados entre trabalhos

Trabalho	Nós	Ganho (%)
Jarrah *et al.* (2009)	725	5.5
Erera *et al.* (2012)	160	4.7
Powell (1986)	140	4.6
Powell *et al.* (1989)	300	2.5
Zhang (2010)	58	11.5
Hellmuth (2004)	18	10.5
Caminhos fixos	33	38
Caminhos por dia da semana	33	38
Sem restrições	33	53

Em especial, deve-se citar o ganho de 38% obtido na heurística com caminhos fixos. Como citado na Seção 5.2, esse modelo é o mais simples de se aplicar a uma operação, já que o uso de rotas fixas já é a prática atual de transportadoras. Sua implantação se limitaria a rever as linhas e a devida preparação das expedições, que é um processo já realizado regularmente pelas transportadoras. Dessa maneira, é possível obter um ganho relevante nos custos de maneira mais ágil, sem a necessidade de mudanças estruturais na operação, o que sugere ser bastante promissor.

Em relação aos outros dois modelos, um fato observado é a diferença encontrada em relação a outros trabalhos. No presente projeto, a redução prevista com o cenário sem restrições de unicidade é bastante relevante, atingindo 53%. Esse ganho é bastante significativo na comparação com a variação de caminhos por dia da semana, o que justificaria a sua implantação. Isso difere das conclusões de Erera *et al.* (2012), em que a abordagem sem restrições apresenta uma redução de custo apenas 20% maior do que o modelo com caminhos por dia da semana.

A diferença nas conclusões pode se dar pela deficiência no resultado do algoritmo com caminhos por dia da semana proposto, que apresentou resultados semelhantes ao próprio caso de caminhos fixos. Como discutido na Seção 5.2, o processo de remoção e reinserção de demandas no caso de caminhos por dia da semana muitas

vezes produz infactibilidades temporais, que diminuem as possibilidades de rota e o espaço de soluções que o algoritmo consegue explorar.

Com uma possível revisão do algoritmo para caminhos por dia da semana no futuro, é possível que o seu resultado se aproxime da solução de caminhos sem restrições, fazendo com que essa se torne menos relevante. Nesse caso, a maior parte do ganho já seria obtida com soluções mais amenas em termos de operação.

Com o resultado observado nesse trabalho, porém, os números sugerem que é bastante válida a implantação de um modelo sem restrições de unicidade de caminhos, uma vez que a sua redução nos custos é bem significativa (53%) frente ao modelo com caminhos fixos, inclusive com diferenças maiores do que foi encontrado em trabalhos similares.

Para essa implantação, conforme mencionado na Seção 5.2, é necessário um investimento nos sistemas de gerenciamento e monitoramento da operação, uma vez que, para cada pacote avaliado, deverá ser feito um mapeamento de qual a sua origem e destino para só então direcionar qual deverá ser a sua expedição e o seu caminho até o destino final.

Essa mudança pode ser bastante estrutural para a maioria das empresas, já que depende de um outro nível de detalhe nos processos de expedição, além de sistemas inteligentes de identificação de pacotes e planejamento, fazendo com que sejam adaptações de médio a longo prazo.

Dado o valor de redução de custos proveniente da mudança de operação, porém, muito provavelmente os custos de investimento serão compensados. Em especial, com o barateamento de tecnologias como RFID que tem sido observado nos últimos anos, as mudanças muito provavelmente seriam justificadas. Além disso, outros benefícios provenientes de um controle mais detalhado nas operações também seriam capturados, como a redução de extravios e a disponibilização em tempo real do posicionamento de uma encomenda, também seriam encontrados com esse tipo de mudança.

Por sua vez, o cenário com caminhos por dia da semana, que envolve uma mudança intermediária nas operações, pelos resultados atuais, não justificaria a sua implantação. Isso porque os ganhos identificados são semelhantes aos do caso de caminhos fixos. Possivelmente, com mudanças no algoritmo ou na aplicação a outra transportadora, essa solução se mostre como uma alternativa interessante para as empresas, como sugerido em outros trabalhos.

Em relação aos tempos de processamento, foi realizada uma avaliação de qual seria o número de iterações adequado para garantir resultados aderentes sem exigir muito em termos de requisitos computacionais (Seção 5.2). Ao processar o modelo para 300 iterações na heurística construtiva e 10 iterações para a busca local, são obtidos os resultados apresentados na Tabela 18, em que também são mostrados resultados em termos de tempo de outros trabalhos.

Como pode ser visto, os tempos de processamento obtidos são bastante similares ao reportado em outros trabalhos, mesmo tratando a variável temporal de maneira contínua. Esse resultado reforça a aplicabilidade dos algoritmos propostos em operações reais de carga fracionada.

Tabela 18: Comparação de tempos de processamento com demais trabalhos na literatura

Trabalho	Nós	Máquina	Processamento (s)
Jarrah *et al.* (2009)	725	3.2GHz/3.6 GB RAM	7.105
Erera *et al.* (2012)	160	2.66GHz/32 GB RAM	21.600
Powell (1986)	140	IBM3081	771
Zhang *etal.* (2010)	58	2.66GHz/4 GB RAM	122
Sem restrições	33	2.2GHz/4 GB RAM	360
Caminhos fixos	33	2.2GHz/4 GB RAM	250
Caminhos por dia da semana	33	2.2GHz/4 GB RAM	322

É possível observar que a abordagem com caminhos fixos é a que requer menor tempo de processamento, seguida dos caminhos por dia da semana e por fim a

abordagem sem restrições. Isso é explicado pelo menor espaço de soluções que o algoritmo tem para percorrer, dado que os caminhos pré-determinados já delimitam quais são os nós que cada demanda deve percorrer. Deve se recordar, porém, que, com caminhos pré-determinados muitas vezes há o reprocessamento de demandas com prazos infactíveis, o que pode aumentar significativamente o tempo necessário para o algoritmo, dependendo da instância tratada.

6 CONCLUSÕES

Ao longo do trabalho, após a apresentação do problema e de uma revisão bibliográfica, foram propostas diferentes modelagens matemáticas para a determinação de linhas de transporte em uma operação de carga fracionada e correspondentes heurísticas construtivas e de buscas locais, variando o tratamento dado à restrição de unicidade de caminhos.

Em relação à modelagem matemática proposta, dada a complexidade do problema, não foi possível aplicar a casos reais, sendo criadas pequenas instâncias geradas aleatoriamente. Ainda assim, o seu uso permitiu comparar os resultados encontrados nas heurísticas, indicando que esses parecem ser uma boa alternativa à identificação de uma solução ótima.

Já ao aplicar as heurísticas propostas no Capítulo 4 a um caso real, é possível afirmar que foram construídos três algoritmos de alocação de linhas de carga fracionada com resultados bastante promissores em termos de aplicação no mercado.

Primeiramente, deve-se discutir a redução de custos potencial identificada, que pode trazer benefícios significativos. Mesmo com simplificações utilizadas na modelagem, como o uso de frota homogênea e capacidade infinita para terminais, que fariam com que nem todo o potencial seja capturado em uma aplicação real, os resultados previstos são promissores.

Em especial, a abordagem sem restrição de unicidade de caminhos apresentou uma redução potencial para o caso aplicado em mais de 50% nos custos totais operacionais, embora seja necessário um maior investimento em termos de adaptação das operações.

Outra abordagem com potencial bastante positivo é o de caminhos fixos, em que a adaptação da operação seria pontual, apresentando uma redução de 38%. A absorção dessa alternativa pelas empresas se daria, portanto, de maneira mais simples, sendo capaz de capturar ganhos a curto prazo, que poderiam, inclusive, financiar investimentos para a posterior implantação do modelo sem restrições de caminhos. Uma das principais preocupações na proposta do algoritmo, o tempo de

processamento, se mostrou bastante similar aos demais trabalhos encontrados na literatura.

A solução proposta para caminhos na semana, por sua vez, não mostrou resultados suficientes para se justificar. O seu ganho frente ao de caminhos fixos é marginal, e a sua implantação em uma operação real seria sensivelmente mais complexa. A revisão do seu algoritmo, em especial na busca local, pode reverter esse cenário no futuro e esse tipo de operação se apresentar como uma boa alternativa para transportadoras, como sugerido em Erera (2012).

Além da potencial redução de custos, uma das maiores contribuições da modelagem matemática e das heurísticas é o tratamento de tempos de maneira contínua. Enquanto outros trabalhos dependem de uma discretização da variável temporal nos seus algoritmos, levando a discussões de qual o nível de detalhe ideal que o tempo deve ser discretizado para gerar resultados aderentes, nesse trabalho todo o tempo é contínuo, evitando questionamentos do gênero. Como discutido na Seção 5.3, mesmo com essa abordagem, os tempos de processamento se mostraram na mesma ordem de grandeza frente a trabalhos similares.

Em especial, o algoritmo sem restrições de unicidade de caminhos se mostrou bastante rápido em termos de processamento. Mesmo processando em uma máquina comum para 300 iterações, os tempos identificados foram de cerca de 6 minutos. Caso o número de iterações seja reduzido para 100 (quando a distância para o mínimo encontrado foi de 1,5% nos cenários processados), esse valor chegaria a 3 minutos.

Com essa velocidade de processamento, alinhada ao maior controle exigido das encomendas ao longo da cadeia, é possível expandir a aplicação do modelo para planejamentos operacionais, e não somente táticos, possibilidade já levantada em Zhang (2010). Sabendo a exata posição de cada pacote e de cada caminhão na rede, além de novos recebimentos feitos em cada terminal de demandas recém chegadas, é possível replanejar com uma frequência diária quais deverão ser os despachos para o dia seguinte.

Nesse cenário, é possível ter uma percepção mais próxima do real em termos de demanda, uma vez que trataria da situação real em termos de demanda e posicionamento de encomendas na cadeia, e não mais com demandas médias. Dessa maneira, ter-se-ia um planejamento muito mais flexível e assertivo, que consegue identificar variações pontuais nas demandas que chegaram na rede e consequentes sinergias e economias que um planejamento tático com demandas mensais médias não é capaz de capturar.

Em relação a possíveis desdobramentos da pesquisa, existem diversos pontos que podem ser tratados e melhorados no futuro. As propostas de algoritmos atuais são capazes de simular de maneira geral qual seria a operação, garantindo a sua factibilidade, porém existem diversos detalhes nas operações que fazem com que sejam realizadas adaptações naturais ao implantar a solução a uma situação real. Isso faria com que nem todo o ganho potencial, portanto, seja capturado, embora seja esperado que a sua ordem de grandeza seja mantida.

Um dos pontos que não são tratados no problema é o transporte dos veículos vazios, que garante o equilíbrio da rede em termos de entrada e saída de veículos, que no trabalho de Hellmuth (2004) correspondem a 10% do total de custos operacionais. A sua tratativa direta no algoritmo pode ser complexa, porém. Uma alternativa que poderia ser implantada é primeiro a resolução do algoritmo de caminhões carregados, e posteriormente o equilíbrio de fluxo de vazios. Dado que esse problema é mais simples, esse processo poderia se dar através de um modelo em programação linear mista.

Uma das principais simplificações do modelo proposto é a consideração de frota homogênea nas operações, que faz com que os próprios dados tenham que ser tratados e simplificados para entrada no modelo (Seção 5.2). Uma possibilidade é considerar a troca de veículo no cálculo do custo marginal no algoritmo de caminhos mínimos. Ao invés de somente considerar o custo associado a abrir uma nova linha de transporte, seria possível aumentar a capacidade de uma já existente, representando a troca por um veículo maior, tendo em contrapartida um maior custo unitário operacional para essa linha.

Também não são consideradas restrições ligadas à capacidade dos terminais em termos de volume tratado diariamente e horário de funcionamento. Em relação ao volume, essa pode ser uma restrição adicionada à checagem da factibilidade de associação de uma demanda a uma perna. Caso o nó de destino tenha a sua capacidade máxima infringida pela adição do volume dessa demanda, essa movimentação seria descartada. Já o horário de funcionamento poderia ser tratado através de tempos de espera nos terminais diferenciados de acordo com o horário que cada veículo chega nos terminais.

Poderão existir ainda outras características específicas da operação de cada empresa, que podem exigir alterações pontuais nos algoritmos, quando considerado válido.

7 REFERÊNCIAS

Ahuja, R., Magnanti, T., Orlin, J. “Network Flows: Theory, Algorithms and Applications”, Prentice Hall: Englewood Cliffs, NJ, 1993.

B. R. Hellmuth, “Uma contribuição ao estudo do planejamento de transporte rodoviário de cargas fracionadas”, 2004. Tese de Mestrado, Escola Politécnica, Universidade de São Paulo, São Paulo, 2004.

Prado, André Alarcon de Almeida.” Modelagem do problema de localização/ roteirização para o transporte de carga fracionada”. 2016. Tese de Doutorado - Escola Politécnica, Universidade de São Paulo, São Paulo, 2016.

Barcos, L., Rodríguez, V., Álvarez, M. J., Robusté, F. “Routing design for less-than-truckload motor carriers using Ant Colony Optimization”. *46*(3), pp. 367–383. 2010.

Barnhart, C., Krishnan, N., Kim, D., Ware K. “Network Design for Express Shipment Delivery”, Volume 21, Issue 3, pp 239–262, 2002.

Crainic, T., “Service network design in freight transportation”, European Journal of Operations Research, vol. 122, no. 2, pp. 272–288, 2000.

Crainic, T., Rousseau, J., “Multicommodity, multimode freight transportation: A general modeling and algorithmic framework for the service network design problem”, Transportation Research Part B: Methodological, 20, issue 3, p. 225-242, 1986

Crainic, T., Roy, J., “OR tools for tactical freight transportation planning”, European Journal of Operational Research, 33, issue 3, p. 290-297, 1988.

Desrochers, M., Savelsbergh, M. W. P., Soumis, F. “Vehicle Routing with Time Windows: Optimization and Approximation”. In Vehicle Routing: Methods and Studies by B. L. Golden and A. A. Assad (eds.). North Holland, Amsterdam, 65- 84, 1988.

Desrochers, M., Soumis, F. “A Generalized Permanent Labeling Algorithm for the Shortest Path Problems with Time Windows”. INFOR 26, 191-212, 1988a.

Desrochers, M., Soumis, F. “A reoptimization algorithm for the shortest path problem with time windows”, European Journal of Operational Research, Volume 35, Issue 2, pp. 242-254, 1988b.

Desrosiers, J., Soumis, F., Desrochers, M. “Methods for routing with time windows, European Journal of Operational Research”, Volume 23, Issue 2, pp. 236-245, 1986.

Erera, A., Karacik, B., Savelsbergh, M. “A dynamic driver management scheme for less-than-truckload carriers”. Computers and Operations Research, vol. 35, issue 11, 3397–3411, 2008.

Erera, A., Hewitt, M., Savelsbergh, M., Zhang, Y. “Improved Load Plan Design Through Integer Programming Based Local Search”. Transportation Science, vol. 47, issue 3, pp. 412–427, 2012.

Erera, A. L., Hewitt, M., Savelsbergh, M. W. P., Zhang, Y. “Creating schedules and computing operating costs for LTL load plans”. Computers and Operations Research, vol. 40, issue 3, pp. 691–702, 2013.

Gendron, B., Crainic, T. G., Frangioni, A. “Multicommodity Capacitated Network Design”. In B. Sansò & P. Soriano (Eds.), Telecommunications Network Planning, pp. 1–19, inbook, Boston, MA: Springer US, 1999.

Gorman, M. “An application of genetic and tabu searches to the freight railroad operating plan problem”. Annals of Operations Research, vol 78, pp. 51–69, 1998.

B. R. Hellmuth, “Uma contribuição ao estudo do planejamento de transporte rodoviário de cargas fracionadas”, São Paulo: Tese de Mestrado, Escola Politécnica da USP, 2004.

Hoppe, B., Klampfl, E.Z., McZeal, C., Rich, J. (1999). “Strategic loadplanning for less-than-truckload trucking”. Center for Research on Parallel Computation Working Paper, Rice University,Houston, Texas, 1999.

Jarrah, A. I., Johnson, E., and Neubert, L. C., “Large-scale, less-than- truckload service network design”, Operations Research, vol. 57, no. 3, pp. 609– 625, 2009.

Katayama, N., Yurimoto, S. "The Load Planning Problem for Less-than-Truckload Motor Carriers and a Solution Approach", Proceedings of the 7th International Symposium on Logistics, pp. 567-572, 2002.

Lin, C. C., & Chen, S. H. “The hierarchical network design problem for time-definite express common carriers”. Transportation Research Part B: Methodological, vol. 38, issue 3, pp. 271–283. 2004

Lin, C.-C., & Chen, S. H. “An integral constrained generalized hub-and-spoke network design problem”. Transportation Research Part E: Logistics and Transportation Review, volume 44, issue 6, pp. 986–1003, 2008.

Powell, W. B., “A local improvement heuristic for the design of less-than- truckload motor carrier networks”, Transportation Science, vol. 20, no. 4, pp. 246–257, 1986

Powell,W. B. and Sheffi, Y., “The load planning problem of motor carriers: Problem description and proposed solution approach”, Transportation Research– A, vol. 17A, no. 6, pp. 471–480, 1983.

Powell,W. B. and Sheffi, Y., “Design and implementation of an interactive optimization system for network design in the motor carrier industry,” Operations Research, vol. 37, pp. 12–29, 1989

Silva, M. Uma contribuição ao projeto de redes de transporte de carga parcelada, 2010. Tese de Doutorado, Escola Politécnica, Universidade de São Paulo, São Paulo, 2010.

Vasco, R. A. & Morabito, R. “The dynamic vehicle allocation problem with application in trucking companies in Brazil”, Computers & Operations Research, vol. 76, 2016, pp. 118-133, 2016.

Zhang Y. “Advances in LTL load plan design” Ph.D. thesis, Georgia Institute of Technology, Atlanta, 2010.