UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL - UFRGS
ESCOLA DE ADMINISTRAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM ADMINISTRAÇÃO

PIETRO TIARAJU GIAVARINA DOS SANTOS

UMA HEURÍSTICA *RELAX-AND-FIX* PARA O *VEHICLE ROUTING PROBLEM WITH PICKUP AND DELIVERY WITH TIME WINDOWS* APLICADO A UM PROBLEMA DE TRANSPORTE MARÍTIMO

PORTO ALEGRE
2019

Pietro Tiaraju Giavarina dos Santos

Uma heurística *relax-and-fix* para o *Vehicle Routing Problem with Pickup and Delivery with Time Windows* aplicado a um problema de transporte marítimo

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre em Administração da Universidade Federal do Rio Grande do Sul - UFRGS, Escola de Administração.

Orientador: Prof. Dr. Denis Borenstein

Porto Alegre
2019

Pietro Tiaraju Giavarina dos Santos

Uma heurística *relax-and-fix* para o *Vehicle Routing Problem with Pickup and Delivery with Time Windows* aplicado a um problema de transporte marítimo

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre em Administração da Universidade Federal do Rio Grande do Sul - UFRGS, Escola de Administração.

Data de Aprovação:

Banca Examinadora

Prof. Dr. Denis Borenstein
Universidade Federal do Rio Grande do Sul - Escola de Administração
Orientador

Prof. Dr. Pablo Cristini Guedes
Universidade Federal do Rio Grande do Sul - Escola de Administração
Examinador

Prof. Dr. Luciano Ferreira
Universidade Federal do Rio Grande do Sul - Escola de Administração
Examinador

Prof. Dr. Olinto César Bassi de Araújo
Universidade Federal de Santa Maria
Examinador

RESUMO

O presente trabalho tem como objetivo apresentar um método de solução para um problema marítimo enfrentado por uma empresa do ramo de fertilizantes químicos. Este problema está associado ao planejamento operacional do processo de coleta de matéria-prima em portos europeus e sua entrega realizada em portos brasileiros para abastecer plantas misturadoras. O problema foi introduzido por Kretschmann (2018), o qual realizou uma formulação matemática baseada no *Vehicle Routing Problem with Pickup and Delivery with Time Windows* (VRPPDTW). Contudo, o metódo de solução desenvolvido por Kretschmann (2018), baseado no *branch-and-bound*, não permitiu a solução dos problemas reais da empresa. O foco deste trabalho concentrou-se no desenvolvimento de um método heurístico baseado em *relax-and-fix* especificamente desenvolvido para resolver instâncias reais. A utilização da heurística permitiu a obtenção de soluções melhores em relação ao sistema vigente empregado pela empresa. Adicionalmente, a heurística reduziu o processo de planejamento de 60 dias para 2 horas em média.

Palavras-chave: Problema marítimo. Metaheurística. Automatização.

ABSTRACT

This study aims to present a solution method for a maritime problem faced by a chemical fertilizer company. This problem is associated to the operational planning of the raw material collection process in European ports and its delivery in Brazilian ports to supply mixing plants. The problem was firstly introduced by Kretschmann (2018), which performed a mathematical formulation based on the Vehicle Routing Problem with Pickup and Delivery with Time Windows (VRPPDTW). Unfortunately, the solution method developed by Kretschmann (2018), based on branch-and-bound, did not allow solving the company's real problems. The focus of this work was the development of a heuristic method based on relax-and-fix specifically designed to solve real instances. The use of heuristics allowed to obtain better solutions in relation to the current system employed by the company. In addition, heuristics reduced the planning process from 60 days to 2 hours on average.

Keywords: Maritime problem. Metaheuristic. Automation.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O desenvolvimento deste trabalho foi motivado pelo problema enfrentado por uma empresa do ramo de fertilizantes químicos, a qual necessita organizar seu procedimento de coleta e entrega de matéria-prima. Estes fertilizantes químicos são produzidos fora do país e são transportados na forma de grânulos por navios graneleiros, coletados em portos na Europa e entregues em diversos portos na costa brasileira. Uma vez no Brasil, os grânulos são transportados por chatas ou caminhões até as plantas, onde serão misturados de acordo com a fórmula dos produtos finais, ensacados e entregues aos produtores rurais, cooperativas de produção ou representantes comerciais.

O processo decisório consiste no planejamento mensal, com no mínimo 60 dias de antecedência, do roteamento e o agendamento das coletas e entregas. Dentre as decisões a serem tomadas, incluem-se: a quantidade de navios e suas rotas a serem percorridas, a alocação dos produtos e suas quantidades nos porões dos navios e o carregamento e descarregamento nos portos com o objetivo final de minimizar o custo do sistema logístico e atender aos prazos e às demandas requeridas (KRETSCHMANN, 2018). Informativamente, pontua-se que esse problema é de caráter tático e aborda apenas uma das etapas da cadeia de suprimentos a qual é submetida.

A empresa dispende de recursos humanos, atualmente, para realizar este processo decisório, o que apresenta ser ineficiente e dispendioso. Dessa forma, relativamente à complexidade do problema, o processo decisório busca uma forma de atender todas as restrições sem uma preocupação com o procedimento de otimização, uma vez que encontrar uma solução viável para problemas desse grau de complexidade apresenta ser tarefa de difícil execução. Dessa forma, fundamenta-se a introdução da Pesquisa Operacional, e consequentemente, da Programação Linear nesta etapa do planejamento para que se possa utilizar um modelo matemático fidedigno ao problema real. Vislumbra-se, assim, uma possível redução de custos multidimensionais no sistema.

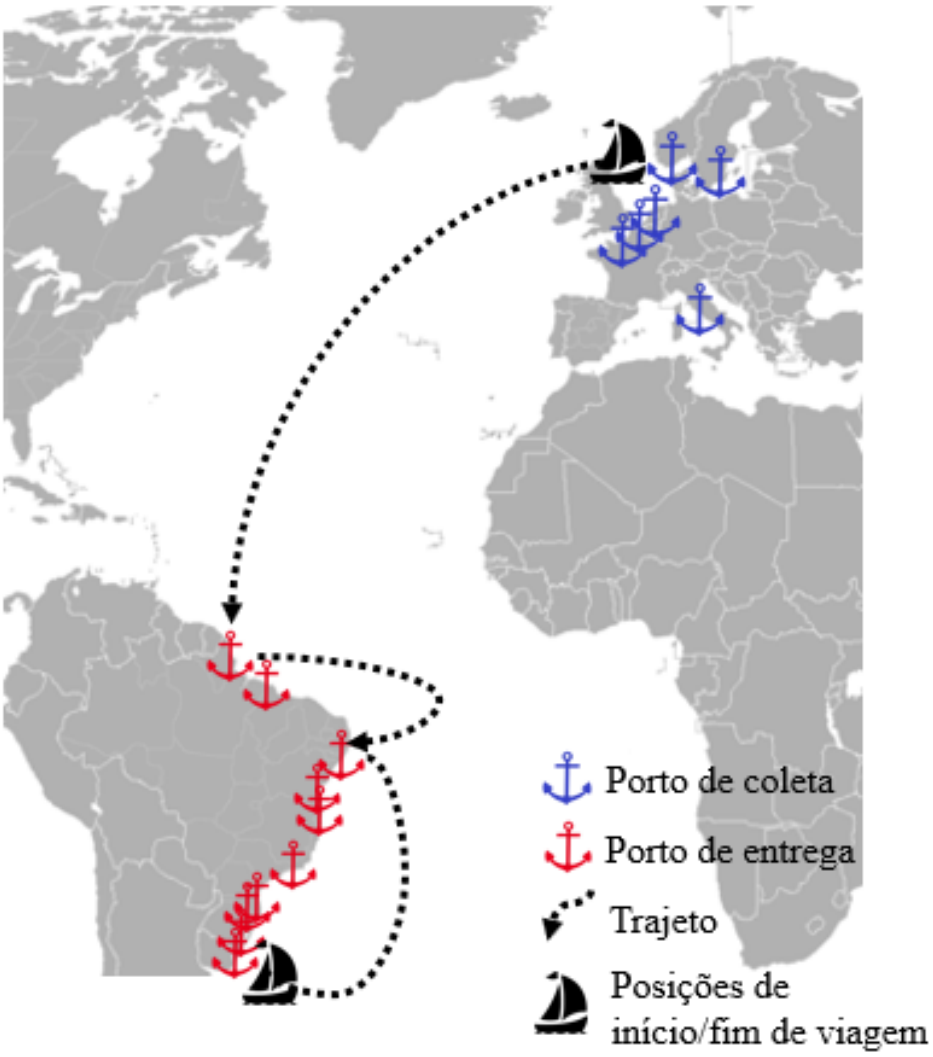
O problema proposto tem como objetivo endereçar as entregas de matérias-primas para as plantas misturadoras no Brasil, a fim de reduzir o custo total desse sistema e proporcionar uma maior margem de lucro na venda dos produtos a serem posteriormente produzidos. O modelo proposto pelo autor, então, visa otimizar uma parte do processo da cadeia de suprimentos, este designado pelo planejamento da coleta e do transporte marítimo de matéria-prima proveniente dos portos europeus até a sua entrega nos portos brasileiros. Nesse sentido, o modelo unifica

tanto as características presentes em problemas derivados do VRPPDTW a fim de consolidar uma solução composta para o problema proposto. As principais características são melhor explicitadas abaixo:

- O roteamento e agendamento de frota é realizado em múltiplos portos, onde esses navios prosseguem primeiramente com o processo de coleta e apenas com o término desta etapa é que se inicia o procedimento de entrega.
- Os portos operam de forma constante: funcionam unicamente como coleta, ou como entrega;
- Os navios designados para o serviço são heterogêneos;
- As janelas de tempo são utilizadas para iniciar e finalizar a execução do serviço prestado em cada porto, seja para carregamento ou descarregamento;
- Os portos só podem ser visitados exclusivamente por um navio e somente por um navio dentro de uma janela de tempo. Caso contrário, é aplicada uma penalização de custo de *demurrage*. Esta penalização ocorre devido ao tempo de espera do navio;
- O transporte marítimo é limitado na variedade de produtos que podem ser transportados;
- Um mesmo navio poderá ter a necessidade de atracar em mais de um porto para realizar alguma operação, seja de coleta ou entrega.

Dessa forma, a função objetivo consiste na minimização dos custos operacionais a partir da resolução do problema de roteamento e agendamento. Na Figura 1 podemos visualizar um exemplo de roteamento para o problema designado, onde a rota de um navio consistiu em visitar um porto de coleta e três portos de entrega.

Figura 1 – Exemplo de roteamento



Fonte: elaborado pelo autor

Enquanto isso, na Tabela 1, podemos visualizar um agendamento possível para o roteamento previamente apresentado, com suas respectivas janelas de tempo.

Tabela 1 – Exemplo de agendamento[1]

Dia	PC I	PE I	PE II	PE III
Dia estimado para chegar	5	27	40	58
Dia estimado para atracar	5	30	41	62
Dia estimado para expedir	9	33	46	65
Dia estimado de disponibilidade do produto	-	36	50	71

Fonte: elaborado pelo autor

Finalmente, pontua-se que as características do *ship scheduling* e do *cargo routing*, problemas derivados do VRPPDTW, foram adequadas para sua aplicação. Desta forma, en-

[1] PC: porto de coleta, PE: porto de entrega.

tão, caracterizando um modelo que envolve a utilização de janelas temporais, roteamento e agendamento de diferentes navios heterogêneos, unicidade de visitas realizadas por cada navio, limitação na quantidade de produtos transportados por um único navio e ocorrência de entrega em múltiplos portos.

Devido ao elevado custo computacional de processamento deste modelo, os otimizadores disponíveis atualmente não são capazes de fornecer alguma solução em tempo hábil dependendo do tamanho e dos parâmetros da instância fornecida. Kretschmann (2018) avaliou, a partir de um conjunto de instâncias reais processado através do otimizador IBM ILOG CPLEX© 12.1.0, que dentre elas, as com maior número de portos, produtos e navios tiveram tempo de processamento bastante elevado a fim de encontrar uma solução viável. Para uma das instâncias, nenhuma resposta foi obtida no intervalo de tempo de 200.000s. Os resultados obtidos por Kretschmann (2018) foram comparados com o processo de tomada de decisão utilizado pela empresa através dos sete exemplos históricos de programação de coletas e entregas de matéria-prima em diferentes meses do ano de 2017. Neste sentido, obteve avanços consideráveis, em que a eficácia econômica média atingida por instância foi de aproximadamente R$ 600.000,00, porém apresentou dificuldades em relação à escalabilidade do problema devido a limitações computacionais, bem como da plataforma de programação utilizada.

Este trabalho está pautado no desenvolvimento de um método de solução para um problema real de navegação de alto teor de complexidade. O método de solução proposto é uma heurística *relax-and-fix*, implementada em linguagem C++ permitindo uma redução dos custos computacionais decorrentes de linguagens interpretadas. A heurística consiste em três etapas principais, as quais são:

- uma etapa construtiva, em que se elabora uma entrada inicial para a etapa principal;
- uma etapa principal, onde a otimização propriamente dita é realizada e;
- uma etapa de pós-processamento, esta com o objetivo de melhorar uma possível solução proveniente da etapa principal.

Adicionalmente, foi desenvolvido um gerador de instâncias aleatórias para o problema em linguagem Python com o intuito de diversificar os experimentos computacionais e comprovar a melhoria em relação aos métodos anteriormente testados, além da escalabilidade da heurística.

2 REVISÃO DA LITERATURA

Historicamente, o *Vehicle Routing Problem* (VRP) é considerado um problema clássico de otimização e pertence a família de problemas $\mathcal{NP}$-difícil. Foi originalmente proposto por Dantzig e Ramser (1959), onde foi designado à distribuição de combustível para estações. Desde então, o tema tem sido amplamente estudado e uma série de variações e adaptações do problema original foram abordadas com o intuito de trazer uma maior aplicabilidade a problemas reais. O *Vehicle Routing Problem with Pickup and Delivery* (VRPPD) é uma generalização do VRP, o qual pertence a uma família conhecida como *Pickup and Delivery Problems* (PDP) e suas aplicações estão relacionadas com o transporte de passageiros e produtos.

Como a maioria das aplicações práticas do VRPPD incluem restrições no tempo em que cada veículo pode visitar um local, então apresenta-se mais conveniente a utilização do modelo com janelas de tempo adicionais, o qual é conhecido como *Vehicle Routing Problem with Pickup and Delivery and Time Windows* (VRPPDTW). Este também pertence à classe de problemas $\mathcal{NP}$-difícil pois generaliza o *Traveling Salesman Problem* (TSP), este também $\mathcal{NP}$-difícil. (BARNHART; LAPORTE, 2006; GAREY; JOHNSON, 2002). Este trabalho fará utilização de um modelo derivado do VRPPDTW desenvolvido por Kretschmann (2018), aplicado a um processo de cadeia de suprimentos realizado em uma empresa do ramo de fertilizantes químicos.

Esta classe de problemas já tem sido investigada por diversos pesquisadores na área de pesquisa operacional (TANIGUCHI et al., 1970). Entretanto, os modelos utilizados eram bastante simplificados e não consideravam as peculiaridades do transporte marítimo, tornando-se um desafio para problemas reais de larga escala. Mesmo assim, técnicas heurísticas foram utilizadas a fim de se obter as melhores soluções dos problemas, como algoritmos genéticos, *simulated annealing* e busca tabu. (KORSVIK; FAGERHOLT, 2010), (TODOSIJEVIĆ et al., 2017).

Dessa forma, o *ship scheduling* se tornou um derivado do VRPPDTW que é aderente à modelagem deste problema. Além disso, o *cargo routing* apresentou importância em certas característica da modelagem. Estes foram inicialmente estudados por Ronen (1983), onde é destilado pontos de relevância para a baixa atenção dada ao *ship scheduling* na época. Descreve-se os principais como: a relativa baixa utilização do meio de transporte em comparação à ferrovias e rodovias; o fato de problemas de *ship scheduling* serem menos estruturados do que os VRPTW tradicionais; o aumento de incerteza com a formação de rotas devido a imprevistos como variações climáticas e problemas mecânicos e; conservadorismo da indústria para implementação de novas ideias. Ronen (1983) ainda criou um método de classificação relativa a essa classe de

problemas.

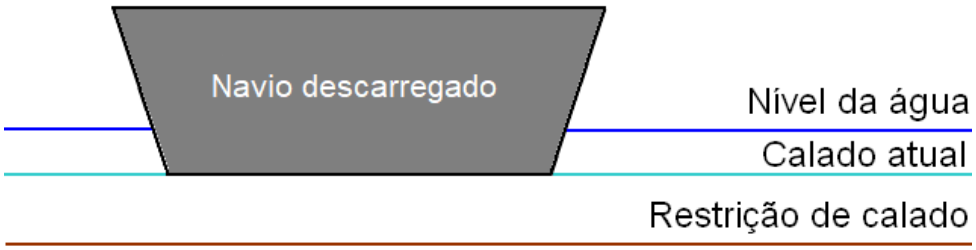
Appelgren (1969) desenvolveu um algoritmo de *column generation* para resolver o problema de *ship scheduling*, podendo-se considerar como o primeiro algoritmo desenvolvido para resolver relativamente a essa classe de problemas. Appelgren (1969) considerou três diferentes formulações e aplicou decomposição de Dantzig-Wolfe em cada uma delas, porém, os modelos utilizados foram bastante simplificados, tendo em geral, apenas restrições referentes ao fluxo, navios e carga.

Christiansen (1999) argumentou que o problema de roteamento e agendamento em transportes terrestres têm sido um tema amplamente discutido na literatura. Em contrapartida, verifica-se que no âmbito marítimo ainda há uma escassez de contribuições expressivas, contrastando com o fato do sistema naval ser o mais utilizado para transporte internacional de cargas. Além disso, o aumento da produção de navios nas últimas décadas é significativo, na época consistindo em mais de 39 mil. O volume de negócios realizados também experimentou um crescimento similar, representando um aumento relativo de 33% entre os anos 90 e 2000 (**??**). Em 2013, 9,6 bilhões de toneladas foram transportadas no mundo sob rotas marítimas, valor que correspondeu a mais de 80% do volume mundial de negócios. (UNCTAD, 2012; UNCTAD, 2014).

Dentre as aplicações analisadas, destacam-se a utilização em transporte de químicos (ARNESEN et al., 2017), graneleiros (AL-KHAYYAL; HWANG, 2007) e contêineres (SONG; DONG, 2012). Papageorgiou et al. (2014) realizou um resumo de diversos trabalhos que utilizaram o MIRP, categorizando-os por aplicação, nível de planejamento, tipo de janela de tempo, modelagem e método de solução, além de um resumo da cardinalidade dos principais *sets* abordados no problema, como número de portos, navios e produtos. Similarmente, Christiansen et al. (2013) promoveram um revisionismo moderno do problema marítimo, mostrando a evolução exponencial do número da frota de navios e do volume de negócios ao longo das décadas. Além disso, o artigo disseca as principais formulações do problema e suas características.

Outra classe de problemas de interesse relativo ao VRPPDTW é o TSP, o qual está diretamente relacionado com o VRP e também apresenta algumas propriedades semelhantes. Rakke et al. (2012) propôs um TSP marítimo com a utilização de *draft limits* (restrições de calado), semelhante ao que ocorre no modelo utilizado neste trabalho. Essas restrições estão exemplificadas na Figura 2.

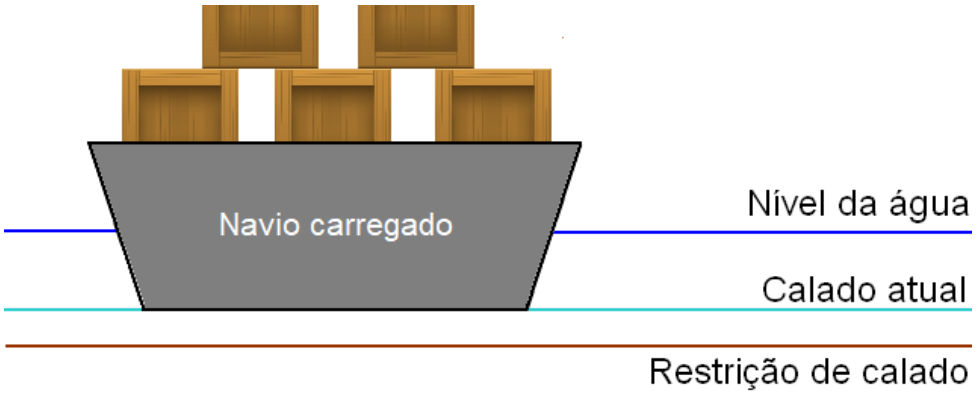
Figura 2 – Exemplo de *draft limit* com um navio descarregado



Fonte: elaborado pelo autor

Nota-se que, com o navio descarregado, uma parte menor do navio fica submersa em comparação a um navio carregado. Essa porção submersa está relacionada com o peso total incidente no navio, bem como a disposição da carga. Se o navio exceder o peso limite da restrição de calado, ele não poderá atracar no porto. A Figura 3 mostra a alteração da posição do calado atual, que se aproxima da restrição de calado do problema pela incidência de peso sobre o navio.

Figura 3 – Exemplo de *draft limit* com um navio carregado



Fonte: elaborado pelo autor

Battarra et al. (2014) estendeu a resolução do problema para múltiplas formulações através de um algoritmo *branch-and-cut* suportado por bibliotecas do CPLEX, as quais também foram utilizadas neste trabalho, em que instâncias difíceis foram resolvidas em sua otimalidade. Na mesma direção, Arnesen et al. (2017) desenvolveu um TSP adicionado de janelas de tempo e *draft limits* para transporte de múltiplos produtos. Ainda, Malaguti et al. (2018) formulou um TSP com *draft limits* e resolveu através de um algoritmo *branch-and-cut* com diversas técnicas de redução de espaço de estados, municiados por um heurística construtiva de inserção. Entretanto,

todos os trabalhos, embora façam o uso dessa multiplicidade de produtos, não compartimentam as cargas, tratando-as como objetos discretos. Procedimentos heurísticos também foram utilizados para resolver essa classe de problemas marítimos, como foi o apresentado por Todosijević et al. (2017) através da técnica de *General Variable Neighborhood Search* (GVNS). Este se fez utilizar de movimentos padrões em heurísticas do TSP, como 2-opt, 3-opt e OR-opt e alcançou a otimalidade para instâncias provenientes da TSPLib. Porém, em geral, os artigos que abordaram o TSP marítimo utilizaram uma formulação genérica da mesma, sem ter uma aplicação real diretamente relacionada com o problema proposto. O modelo proposto neste trabalho tem a vantagem de abordar uma situação real do planejamento empresarial, esta de caráter tático.

Brønmo et al. (2007) introduziram o problema de *ship scheduling* com tamanhos de carga variáveis formulado através de uma abordagem voltada ao *set partitioning*. Agarwal e Ergun (2008) apresentaram um modelo misto de *ship scheduling* e *cargo routing* que considerou na modelagem a frota através de conjuntos de tipos de navios, flexibilizando as combinações de navios a serem utilizado numa solução. Song e Dong (2012) desenvolveram um complexo modelo de *cargo routing* com diversas variáveis binárias relativas à transbordo e expedição. Além disso, propôs restrições adicionais o problema como por exemplo, o número de contêineres vazios que estão esgotados em um tempo k. Ainda, é possível observar na literatura a utilização do *ship scheduling* em problemas de caráter misto, como no roteamento simultâneo naval e aéreo proposto por Aytekin (2002) e Li et al. (2007).

Em linhas gerais, a contribuição deste trabalho está:

- na introdução de uma formulação MILP, que se trata de uma novidade para o *ship routing* e o *ship scheduling*;
- no desenvolvimento de uma heurística capaz de oferecer soluções em tempo razoável para instâncias grandes do problema;
- no oferecimento de um conjunto de instâncias geradas aleatoriamente para o problema que poderão ser utilizados para trabalhos futuros;
- na demonstração através de um estudo computacional que a abordagem via otimização pode ser utilizado com grandes benefícios para o planejamento operacional da empresa.

3 METODOLOGIA

3.1 FORMULAÇÃO

Este modelo foi integralmente proposto por Kretschmann (2018), onde poderá ser obtido mais detalhes sobre a aplicação, formulação e restrições. Em geral, o problema do VRPPDTW pode ser tratado como uma rede de fluxo. O grafo $G = (N, A)$ é formado pelos nós N, que representam os portos de coleta, os portos de entrega e os armazéns artificiais, necessários para o equilíbrio do sistema e pelos arcos A, que representam os possíveis caminhos a serem percorridos pelos navios. A Figura 4 esquematiza um exemplo de grafo com quatro portos de coleta e quatro portos de entrega.

É possível perceber a ausência de arcos de entrada nos portos de coleta (N^P) proveniente dos portos de entrega (N^D), uma das restrições de operacionalidade do problema. Além disso, não há arcos de entrada proveniente dos portos de entrega (N^D) no armazém artificial s, bem como não há arcos de entrada no armazém artificial f proveniente dos portos de coleta (N^P), estas restrições como garantia do equilíbrio da rede de fluxo. Como premissa do problema, é necessário que um navio visite, no mínimo, um nó pertencente ao conjunto N^P e um nó pertencente ao conjunto N^D, garantindo a ocorrência do procedimento de *pickup* e *delivery*, respectivamente.

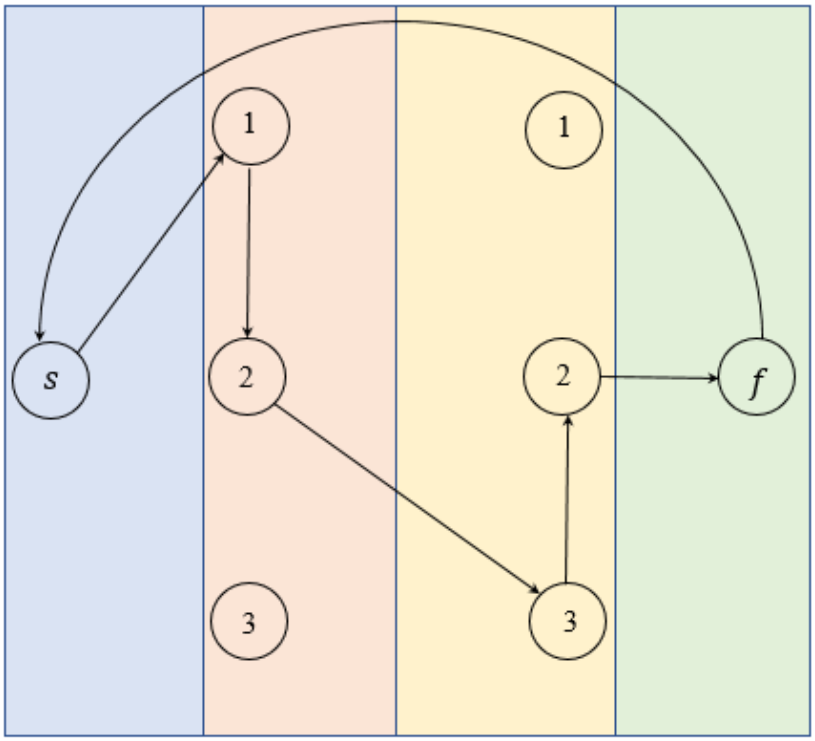
Figura 4 – Exemplo de grafo do problema

Fonte: elaborado pelo autor

Portanto, cada grafo do tipo $G_v = (N_v, A_v)$ irá determinar um roteamento designado para um navio específico, onde $N_v = N_v^P \cup N_v^D \cup s \cup f$. Cada grafo G_v será formado pelos nós N_v, que representam os portos e os armazéns artificiais a serem visitados pelo navio v e pelos arcos A_v, que são os caminhos percorridos pelo navio v. Arcos do tipo (N^P, N^D) podem ser utilizados apenas uma única vez em cada grafo G_v.

A Figura 5 representa um grafo G_v, solução possível de um navio v para a rede de fluxo representada na Figura 4.

Figura 5 – Exemplo de solução de viagem determinada para um navio



Fonte: elaborado pelo autor

3.2 MODELO

Sejam os índices abaixo, onde:

- i, k representa o porto de origem;
- j representa o porto de destino;
- h representa o porto de origem ao de origem;
- p representa o produto;
- v, w representa o navio.

Sejam os conjuntos abaixo, onde:

- N^V representa o conjunto dos portos navegáveis, indexado em i, j e h;
- N^P representa o conjunto dos portos de coleta, indexado em i, j e h;
- N^D representa o conjunto dos portos de entrega, indexado em i, j e h;

- N representa o conjunto dos portos em união com armazéns artificiais de início e fim, indexado em i, j e h;
- P representa o conjunto dos tipos de produtos, indexado por p;
- V representa o conjunto dos navios, indexado por v.

Sejam as variáveis de decisão abaixo, onde:

- l_{ijpv} representa o total da carga, em toneladas, por produto p transportado do porto i ao porto j pelo navio v;
- eta_{ijv} representa o tempo estimado de chegada, em dias, do navio v ao porto j, proveniente do porto i;
- ets_{ijv} representa o tempo estimado de saída, em dias, do navio v do porto j, que chegou ao porto i;
- $etud_{ijpv}$ representa o tempo estimado de atraso ou adiantamento, em dias, que a carga do produto p chegou ao porto j, proveniente do porto i pelo navio v;
- x_{ijv} representa a variável binária de fluxo que atribui a utilização ou não do arco que conecta o porto i ao porto j, pelo navio v;
- y_{ijpv} representa a variável binária que indica se o produto p foi transportado do porto i ao porto j, pelo navio v;
- yd_{ijpv} representa a variável binária que indica se o produto p foi descarregado no porto j, proveniente do porto i, pelo navio v;
- g_{pv} representa a variável binária que indica o carregamento do produto p pelo navio v;
- aux_{vw} representa a variável binária de apoio para a restrição de dois navios não utilizarem o mesmo porto ao mesmo tempo.

Sejam os parâmetros abaixo, onde:

- Q_{ip} representa a carga, em toneladas, do produto p em estoque (se positivo) ou em demanda (se negativo) no porto i;
- TD_{jp} representa a data solicitada de entrega, em dias, do produto p no porto j;

- TDA_j representa o tempo máximo permitido, em dias, do adiantamento da carga solicitada no porto j;
- TDL_j representa o tempo máximo permitido, em dias, do atraso da carga solicitada no porto j;
- TS_{ij} representa o tempo de travessia, em dias, entre os portos i e j;
- TO_j representa o tempo de operação, em dias, para carregamento e descarregamento no porto j;
- TC_j representa o tempo de *clearance*, em dias, dos produtos no porto j;
- TQ_j representa o tempo de fila, em dias, estimado no porto j;
- TPS_j representa a data inicial da janela de tempo, em dias, de coleta no porto j;
- TPF_j representa a data final da janela de tempo, em dias, para coleta no porto j;
- KP_j representa a capacidade de carga máxima, em toneladas, que o porto j está apto a receber ou despachar, por restrições físicas do calado;
- $KVMAX_v$ representa a capacidade de carga máxima, em toneladas, que o navio v está apto a transportar;
- $KVMIN_v$ representa a capacidade de carga mínima, em toneladas, que o navio v está apto a transportar;
- $KVPMIN_v$ representa a capacidade de carga mínima, em toneladas, que o navio v pode transportar de cada produto;
- G_v representa a capacidade máxima, em número de porões, de diferentes produtos a serem transportados pelo navio v, que variam conforme a capacidade de diferentes porões no navio (uma vez que os produtos não podem ser misturados no mesmo porão);
- CD_v representa o custo diário, em reais (R$), do custo diário de *demurrage* pago ao operador do navio v de acordo com o tempo em fila no porto;
- CP_i representa o custo fixo, em reais (R$), de utilizar o porto i;
- CV_v representa o custo diário, em reais (R$), de fretamento do navio v;

- CUD representa o custo diário, em reais (R$), de atraso ou antecipação da entrega da carga no porto de destino;
- CSN representa o custo, em reais (R$), por capacidade máxima de transporte do navio por inverter a rotação geográfica da rota do navio de norte-sul para sul-norte;
- VPL_v representa o número limite de paradas, em quantidade de portos, que o navio v pode fazer nos portos de coleta;
- VDL_v representa o número limite de paradas, em quantidade de portos, que o navio v pode fazer nos portos de descarga;
- PVL_j representa o número limite de navios, em quantidade de navios, que o porto j pode receber para coleta;
- SN_{ij} representa a matriz binária que indica uma rota de orientação sul-norte entre os portos i e j.

Seja a definição especial abaixo, onde:

- $\mathcal{M}$ representa um número suficientemente grande.

Seja a função objetivo, representada por:

$$\begin{aligned}\min \sum_{v\in V} CV_v \left(\sum_{i\in N^D} eta_{i\{f\}v} + \sum_{j\in N^P} eta_{\{s\}jv} \right) + \sum_{i\in N}\sum_{j\in N^V}\sum_{v\in V} CP_j x_{ijv} \\ + \sum_{i\in N}\sum_{j\in N^V}\sum_{v\in V} CD_v TQ_j x_{ijv} + \sum_{i\in N^V}\sum_{j\in N^D}\sum_{p\in P}\sum_{v\in V} CUDetud_{ijpv} \\ + \sum_{i\in N^V}\sum_{j\in N^V}\sum_{v\in V} SN_{ij} CSN_v MAX_v x_{ijv}\end{aligned} \tag{1}$$

A função objetivo (1) minimiza o custo de fretamento, o custo de utilização dos portos, os custos de *demurrage*, a penalidade por desrespeitar as janelas de tempo de entrega e a penalidade pela inversão do sentido geográfico da rota. Estes custos são computados da mesma maneira em que é executada na empresa.

Não obstante, está sujeito às restrições:

$$Q_{ip} - \sum_{j\in N^V}\sum_{v\in V} l_{ijpv} + \sum_{h\in N^V}\sum_{v\in V} l_{hipv} \geq 0 \qquad \forall i \in N^V,\ \forall p \in P \tag{2}$$

As Restrições (2) garantem que a oferta e a demanda de cada produto nos portos de coleta e entrega, respectivamente, são respeitados.

$$\sum_{j\in N}\sum_{p\in P} l_{ijpv} \leq KP_i \qquad \forall i \in N^V,\ \forall v \in V \tag{3}$$

$$\sum_{j\in N}\sum_{p\in P} l_{ijpv} \leq KP_j \qquad \forall j \in N^V,\ \forall v \in V \tag{4}$$

As Restrições (3) e (4) asseguram que um navio está entrando ou saindo de um porto respeitando os limites máximos de calado.

$$Q_{ip} - \sum_{j \in N^V} l_{ijpv} + \sum_{h \in N^P} l_{hipv} \geq 0 \qquad \forall i \in N^P,\ \forall p \in P,\ \forall v \in V \tag{5}$$

$$\sum_{h \in N^V} l_{hipv} - \sum_{j \in N^D} l_{ijpv} \geq 0 \qquad \forall i \in N^D,\ \forall p \in P,\ \forall v \in V \tag{6}$$

$$\sum_{i \in N^P} \sum_{j \in N^D} \sum_{p \in P} l_{ijpv} \leq MAX_v \qquad \forall v \in V \tag{7}$$

$$\sum_{i \in N^P} \sum_{j \in N^D} \sum_{p \in P} l_{ijpv} \geq \sum_{j \in N^V} MIN_v x_{sjv} \qquad \forall v \in V \tag{8}$$

$$l_{ijpv} \geq PMIN_{pv} y_{ijpv} \qquad \forall i \in N^P,\ \forall j \in N^V,\ \forall p \in P,\ \forall v \in V \tag{9}$$

$$\sum_{p \in P} g_{pv} \leq G_v \qquad \forall v \in V \tag{10}$$

As Restrições (5) garantem que um navio não possa sair de um porto de coleta com um produto que não está armazenado. Similarmente, as Restrições (6) não permitem que um navio descarregue uma quantidade maior de um produto nos portos de entrega do que a sua carga corrente. As Restrições (7) garantem que a capacidade máxima de cada navio é respeitada. As Restrições (8) asseguram que nenhum navio é sub-utilizado, carregando dos portos de coleta ao de entrega uma quantidade menor de produtos do que o parâmetro $KVPMIN_v$. As Restrições (9) definem a quantidade mínima de cada produto a ser transportada por um navio. As Restrições (10) limitam o número de diferentes produtos a serem transportados pelo número de compartimentos disponíveis em cada navio.

$$\sum_{j \in N} x_{ijv} \leq 1 \qquad \forall i \in N,\ \forall v \in V \tag{11}$$

$$\sum_{i \in N} x_{ijv} \leq 1 \qquad \forall j \in N,\ \forall v \in V \tag{12}$$

$$\sum_{j \in N} x_{ijv} - \sum_{h \in N} x_{hiv} = 0 \qquad \forall i \in N,\ \forall v \in V \tag{13}$$

$$\sum_{i \in N^P} \sum_{j \in N^D} x_{ijv} \leq 1 \qquad \forall v \in V \tag{14}$$

$$x_{ijv} = 0 \qquad \forall i \in N^D,\ \forall j \in N^P,\ \forall v \in V \tag{15}$$

$$x_{fsv} = 0 \qquad \forall v \in V \tag{16}$$

$$x_{fjv} = 0 \qquad \forall j \in N^V,\ \forall v \in V \tag{17}$$

$$x_{isv} = 0 \qquad \forall j \in N^V,\ \forall v \in V \tag{18}$$

$$\sum_{p \in P} l_{ijpv} - M x_{ijv} \leq 0 \qquad \forall i, j \in N,\ \forall v \in V \tag{19}$$

$$l_{ijpv} - M g_{pv} \leq 0 \qquad \forall i, j \in N,\ \forall p \in P,\ \forall v \in V \tag{20}$$

$$l_{ijpv} - M y_{ijpv} \leq 0 \qquad \forall i, j \in N,\ \forall p \in P,\ \forall v \in V \tag{21}$$

As Restrições (11) e (12) limitam que qualquer arco (i, j) seja utilizado mais de uma vez pelo mesmo navio. As Restrições (13) são relativas à conservação do fluxo. As Restrições (14) estabelecem a conexão entre os portos de coleta e entrega, enquanto as Restrições (14) previnem a existência de arcos conectando os portos de entrega aos de coleta. As Restrições (16), (17) e (18) delimitam as possibilidades dos arcos conectando os armazéns artificiais s e t. As Restrições (19) e (20) definem que o fluxo da quantidade um produto p no arco (i, j) transportado por um navio v apenas ocorre se o navio percorre o arco (i, j), carregando um produto p em algum de seus porões, respectivamente. As Restrições (21) conectam a variável contínua l_{ijpv} à sua contrapartida binária, y_{ijpv}.

$$eta_{ijv} \geq TPS_j x_{ijv} \quad \forall i \in N,\ \forall j \in N^P,\ \forall v \in V \tag{22}$$

$$etd_{ijv} \leq TPF_j x_{ijv} \quad \forall i \in N,\ \forall j \in N^P,\ \forall v \in V \tag{23}$$

$$etd_{ijv} \geq eta_{ijv} + TQ_j + TO_j + M(x_{ijv} - 1) \quad \forall i \in N,\ \forall j \in N^V,\ \forall v \in V \tag{24}$$

$$eta_{ijv} \geq etd_{hiv} + TS_{ijv} + M(x_{ijv} - 1) \quad \forall h \in N,\ \forall i \in N^V,\ \forall j \in N^V,\ \forall v \in V \tag{25}$$

$$etud_{ijpv} \geq \mid TD_{jp} - TC_j - etd_{ijv} \mid + M(yd_{ijpv} - 1) \quad \forall i \in N^V,\ \forall j \in N^D,\ \forall p \in P,\ \forall v \in V \tag{26}$$

$$eta_{ifv} \geq etd_{hiv} + M(x_{i\{f\}v} - 1) \quad \forall i, h \in N^V,\ \forall p \in P,\ \forall v \in V \tag{27}$$

$$eta_{siv} \leq eta_{ijv} - TS_{ij} - TO_i - TQ_i - M(x_{ijv} - 1) \quad \forall i, j \in N^V,\ \forall p \in P,\ \forall v \in V \tag{28}$$

$$eta_{ijv} + TQ_j - M(x_{ijv} - 1) + Maux_{vw} \geq etd_{kjw} + M(x_{kjw} - 1) \quad \forall i, k \in N,\ \forall j \in N^V,\ \forall v, w \in V, v \neq w \tag{29}$$

$$eta_{kjw} + TQ_j - M(x_{kjw} - 1) + M(1 - aux_{vw}) \geq etd_{ijv} + M(x_{ijv} - 1) \quad \forall i, k \in N,\ \forall j \in N^V,\ \forall v, w \in V, v \neq w \tag{30}$$

$$etd_{ijv} \leq TD_{jp} + TDL_j - TC_j + M(-yd_{ijpv} + 1) \quad \forall i \in N^V,\ \forall j \in N^D,\ \forall p \in P,\ \forall v \in V \tag{31}$$

$$etd_{ijv} \geq TD_{jp} - TDA_j - TC_j - M(yd_{ijpv} - 1) \quad \forall i \in N^V,\ \forall j \in N^D,\ \forall p \in P,\ \forall v \in V \tag{32}$$

As Restrições (22) e (23) são relacionadas às janelas de tempo na coleta dos produtos,

que afetam os tempos de chegada e saída, respectivamente, de cada navio para cada porto. As Restrições (24) definem que o tempo de saída de cada navio proveniente de um porto depende do seu tempo de chegada, tempo estimado de fila e tempo de carregamento/descarregamento no porto. As Restrições (25) determinam que o tempo de chegada no porto j de cada navio como relacionado ao tempo de partida do porto i adicionado do tempo de travessia do arco (i, j) pelo navio v, TS_{ijv}. As Restrições (26) descrevem uma chegada adiantada ou uma saída atrasada de um navio em cada porto. As Restrições (27) e (28) definem o tempo inicial e final de roteamento de cada navio. As Restrições (29) e (30) garantem que dois navios não podem ocupar um porto ao mesmo tempo. As Restrições (31) e (32), similarmente a (22) e (23), estabelecem a janela de tempo de partida de cada navio proveniente de um porto, considerando o dia da demanda do produto no porto, TD_{jp}.

$$\sum_{i \in N^P} \sum_{j \in N^P} x_{ijv} \leq VPL_v - 1 \qquad \forall v \in V \tag{33}$$

$$\sum_{i \in N^D} \sum_{j \in N^D} x_{ijv} \leq VDL_v - 1 \qquad \forall v \in V \tag{34}$$

$$\sum_{i \in N} \sum_{v \in V} x_{ijv} \leq PVL_j \qquad \forall j \in N^V \tag{35}$$

$$y_{ijpv} - x_{ijv} \leq 0 \qquad \forall i, j \in N,\ \forall p \in P,\ \forall v \in V \tag{36}$$

$$y_{ijpv} - g_{pv} \leq 0 \qquad \forall i, j \in N,\ \forall p \in P,\ \forall v \in V \tag{37}$$

$$etd_{ijv} - Mx_{ijv} \leq 0 \qquad \forall i, j \in N,\ \forall v \in V \tag{38}$$

$$eta_{ijv} - Mx_{ijv} \leq 0 \qquad \forall i, j \in N,\ \forall v \in V \tag{39}$$

$$y_{ijpv} - TD_{jp} yd_{ijpv} \leq 0 \qquad \forall i \in N^V,\ \forall j \in N^D,\ \forall p \in P,\ \forall v \in V \tag{40}$$

As Restrições (33) e (34) limitam o número de portos de coleta e entrega, respectivamente, que cada navio pode atracar em sua rota. As Restrições (35) limitam o número de navios que poderão atracar em cada porto durante o horizonte de planejamento. As Restrições (36) e (37) estipulam que cada navio pode transportar apenas um produto utilizando o arco (i, j) se: (i) o navio utiliza o arco (i, j) em sua rota, e (ii) se o navio está carregando o produto em um de seus porões. As Restrições (39) e (38) estipulam que cada navio possui um tempo de chegada e saída, respectivamente, de um porto j oriundo de um porto i, se utiliza um arco (i, j) em sua rota. Finalmente, as Restrições (40) garantem que cada produto é apenas descarregado de um navio em um porto se: (i) existe uma demanda pelo produto naquele porto, e (ii) se o navio está carregando aquele produto na sua rota.

$$l_{ijpv} \geq 0 \qquad \forall i, j \in N,\ \forall p \in P,\ \forall g \in V \tag{41}$$

$$etd_{ijv} \geq 0 \qquad \forall i, j \in N,\ \forall v \in V \tag{42}$$

$$eta_{ijv} \geq 0 \qquad \forall i, j \in N,\ \forall v \in V \tag{43}$$

$$etud_{ijpv} \geq 0 \qquad \forall i, j \in N,\ \forall p \in P,\ \forall v \in V \tag{44}$$

$$x_{ijv} \in \{0, 1\} \qquad \forall i, j \in N,\ \forall v \in V \tag{45}$$

$$y_{ijpv} \in \{0, 1\} \qquad \forall i, j \in N,\ \forall p \in P,\ \forall v \in V \tag{46}$$

$$yd_{ijpv} \in \{0, 1\} \qquad \forall i, j \in N,\ \forall p \in P,\ \forall v \in V \tag{47}$$

$$g_{pv} \in \{0, 1\} \qquad \forall p \in P,\ \forall v \in V \tag{48}$$

$$aux_{vw} \in \{0, 1\} \qquad \forall v, w_{v \neq w} \in V \tag{49}$$

A formulação do problema é claramente $\mathcal{NP}$-difícil e, sem a utilização de um método heurístico, conseguir-se-á resolver apenas problemas pequenos em tempo aceitável. Esse aspecto motivou o desenvolvimento de uma heurística para resolver instâncias reais, as quais Kretschmann (2018) teve dificuldade para encontrar soluções em tempo competitivo.

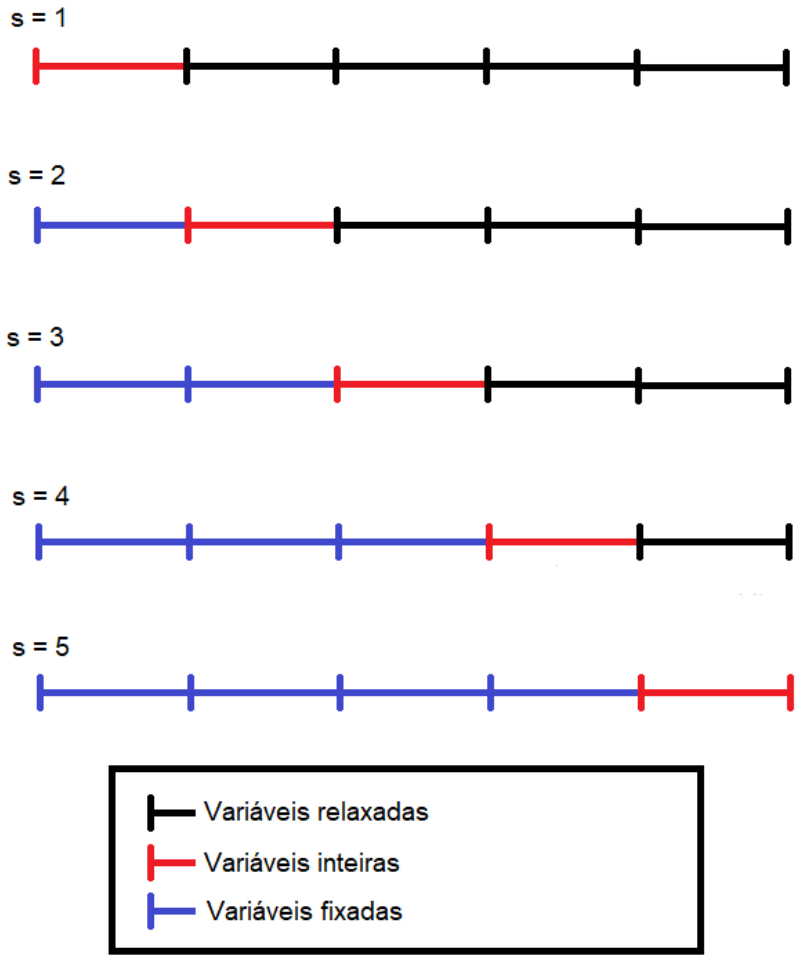
4 Heurística

A heurística funciona através de um procedimento composto por uma fase inicial, cinco fases de aplicação do *relax-and-fix* e duas fases de aprimoramento e possui inspiração no trabalho realizado em Ferreira et al. (2010), especialmente na implementação do algoritmo. Entretanto, a principal diferença se situa na não-divisão das variáveis em conjuntos disjuntivos e devido a características inerentes da formulação deste problema, fizeram com que a abordagem utilizada fosse através do rolamento dos conjuntos de variáveis já existentes no problema.

O *relax-and-fix* é uma heurística que apareceu como suporte à resolução de problemas de pesquisa operacional no início dos anos 2000, encontrado problema misto marítimo-aéreo apresentado por Aytekin (2002) e no problema de agendamento (KELLY; MANN, 2004) e (ESCUDERO; SALMERON, 2005). Entretanto, seu uso ficou bastante conhecido nos problemas de *lot-sizing* (BERALDI et al., 2006), (BERALDI et al., 2008) e (MOHAMMADI et al., 2010).

Técnicas de *relax-and-fix* desenvolvidas para o MIRP foram ainda menos exploradas, conforme melhor visualizado em Papageorgiou et al. (2014). A ideia básica do método é dividir o horizonte de planejamento em um número finito de intervalos n. Dillenberger et al. (1994) explicou que esse problema é decomposto em n sub-problemas e resolvido em iterações que correspondem às janelas de tempo. Uggen et al. (2013) define um conjunto de etapas principais de uma estratégia *relax-and-fix* genérica, em que, inicialmente, a restrição de integralidade de todas as variáveis inteiras é removida. Na primeira iteração, dado um contador s, faz-se $s = 1$. O sub-problema é resolvido com as variáveis inteiras correspondentes ao primeiro intervalo, enquanto as variáveis remanescentes permanecem contínuas e as variáveis inicialmente contínuas do problema são inalteradas. Nas iterações subsequentes, então, no lugar das variáveis previamente inteiras, são fixadas com os valores de uma solução obtida na etapa anterior. Um novo intervalo recebe a condição de integralidade e as demais variáveis permanecem contínuas e não-fixadas. Esse processo é repetido até $s = n$. Após resolver a iteração n, uma solução completa para o problema original é encontrada. Tem-se então, de forma concisa, a operacionalidade de três blocos distintos de variáveis, sendo eles: bloco fixo, bloco inteiro e bloco contínuo. Na Figura 6 tem-se um exemplo da evolução temporal do algoritmo, com detalhe para cada iteração s.

Figura 6 – Exemplo de estratégia *relax-and-fix*



Fonte: elaborado pelo autor

Ferreira et al. (2010) utilizou o *relax-and-fix* em um modelo de *lot scheduling*, onde foi estudado estratégias de implementação e resolução com o suporte do CPLEX, chamado lá de otimizador de próposito geral. Dessa forma, foram definidas estratégias de utilização baseadas em uma análise do problema. Dentre elas, destacam-se o pré-processamento, geração de *cutting planes* e definição de *primal bounds* bem como a ordem de fixação das variáveis inteiras.

A abordagem tratada por Friske e Buriol (2018) buscou uma melhora de *bounds* através da utilização de *valid inequalities* para a formulação *arc-flow* do MIRP. A técnica de *relax-and-fix* proposta é similar a descrita em Uggen et al. (2013) também utilizando os horizontes de tempo como os intervalos do algoritmo. Fez valer-se de uma fase de melhoramento pós-processamento através de procedimentos de busca local. Em geral, o *relax-and-fix* é operado através do "rolamento" sobre as restrições de janelas de tempo devido a questões relacionadas à formulação do problema, onde a janela de tempo é incluída como índice das variáveis binárias, e, portanto, acaba sendo pertinente. O modelo proposto por Kretschmann (2018) é diferenciado nesse aspecto por tratar as janelas de tempo como contínuas e de forma independente, em relação

às variáveis binárias do problema. Esta formulação indica uma possibilidade de alteração da característica mais usual do *relax-and-fix*, através a utilização de classes de variáveis.

Dessa forma, a técnica *relax-and-fix* proposta trata os conjuntos de variáveis existentes no problema como blocos, em que cada etapa um conjunto adicional de variáveis recebe as restrições de integralidade. Como o problema possui cinco grupos de variáveis inteiras, a fase principal do algoritmo possui cinco etapas, relativas à cada um dos procedimentos que serão visualizados a seguir.

4.1 ETAPAS DO MÉTODO HEURÍSTICO

O método de solução proposto é uma heurística *relax-and-fix* que se faz valer da utilização de bibliotecas disponíveis para a linguagem C++ do IBM ILOG CPLEX, suportada por uma etapa construtiva e um gerador de instâncias aleatórias, ambas desenvolvidas em linguagem Python.

As etapas do método dispõem-se da seguinte forma:

- Etapa construtiva:

  I. *Pickup*.

  II. *Delivery*.

- Etapa de aplicação do *relax-and-fix*:

  I. x_{ijv} inteiro.

  II. x_{ijv}, g_{pv} inteiros.

  III. x_{ijv}, g_{pv}, y_{ijpv} inteiros.

  IV. x_{ijv}, g_{pv}, y_{ijpv}, yd_{ijpv} inteiros.

  V. x_{ijv}, g_{pv}, y_{ijpv}, yd_{ijpv}, aux_{vw} inteiros.

- Etapa de pós-processamento:

  I. Polimento.

  II. Reparação do polimento.

  Estas serão melhor detalhadas nas seções seguintes.

4.1.1 Etapa construtiva

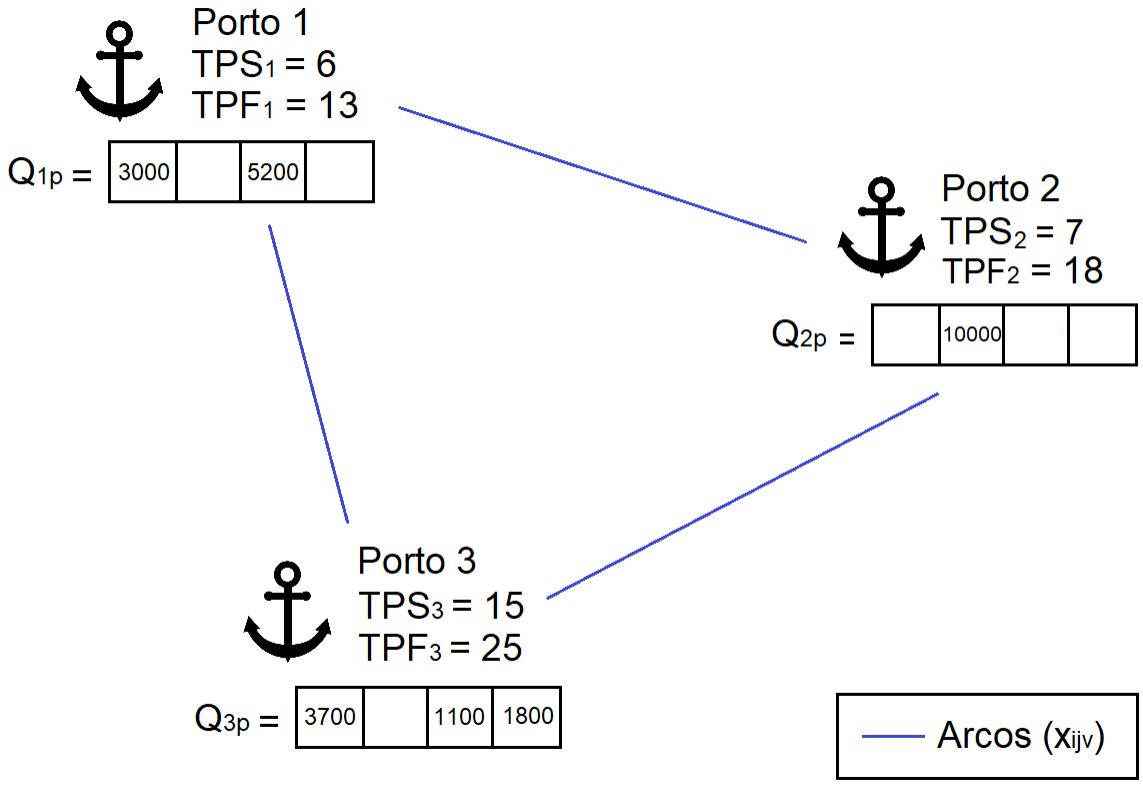
A heurística faz-se valer de uma etapa inicial optativa que gera uma tentativa de solução para aquecer a primeira etapa da fase de aplicação do *relax-and-fix*. O objetivo é gerar uma possível solução inicial, ou uma pista significativa da solução a qual facilitará a busca do otimizador na fase inicial.

Nesta etapa, primeiramente é definida a entrada de dados do problema. Carrega-se os dados que são fornecidos da instância na memória da heurística.

4.1.1.1 *Pickup*

Inicialmente, o porto com menor valor $\frac{TPS_i+TPF_i}{2}$ é selecionado. O primeiro navio então é alocado com o máximo de produtos que ele puder carregar em relação à sua capacidade remanescente em toneladas e o número de porões disponíveis. Na iteração subsequente, o navio será direcionado para o porto de coleta mais próximo que ainda houver produtos que ele possa carregar. O critério de parada ocorre quando não houver mais produtos nos portos de coleta para o navio coletar, o navio não tiver mais porões disponíveis para alocar novos produtos, o navio estiver no limite da capacidade máxima permitida ou ainda se o navio tiver visitado um número limite de portos de coleta permitido. Um exemplo de roteamento de *pickup* será mostrado a seguir.

Figura 7 – Exemplo de estrutura de portos de coleta

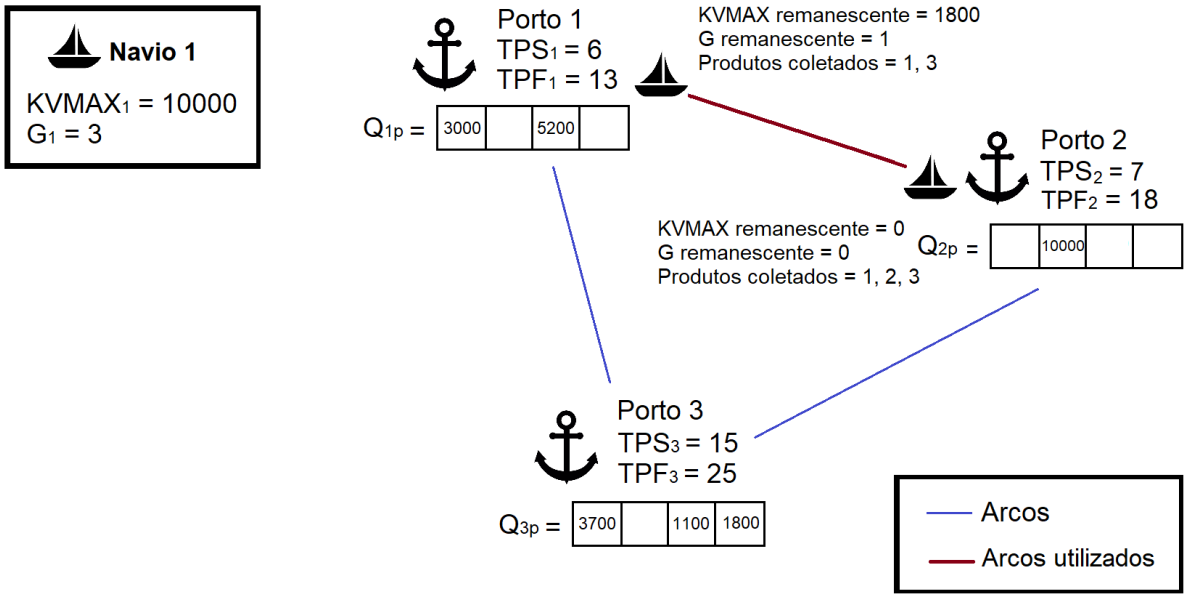


Fonte: elaborado pelo autor

Na Figura 7, temos três portos de coleta equidistantemente dispostos, com seus respectivos TPS_i, TPS_j e Q_{ip} bem como três navios disponíveis para realizar as coletas, estes com $KVMAX_1 = 10000$ e $G_1 = 3$. Utilizando o critério disposto anteriormente para selecionar o primeiro porto, temos que $\frac{TPS_1+TPF_1}{2} = 9.5$, $\frac{TPS_2+TPF_2}{2} = 12.5$ e $\frac{TPS_3+TPF_3}{2} = 20$. Logo, o algoritmo irá selecionar o Porto 1 para o Navio 1 atracar. O Navio 1 coletará o vetor de oferta correspondente aos produtos disponíveis no Porto 1, a saber, $Q_{1p} = [3000, 0, 5200, 0]$. Atuali-

zando suas propriedades, o número de porões disponíveis no Navio 1 após a primeira coleta é de apenas 1, e a $KVMAX_1$ remanescente é de 1800. Como há ainda capacidade disponível bem como porões remanescentes, o Navio 1 irá realizar mais uma coleta em outro porto. De acordo com o algoritmo, o próximo porto a ser atracado é o Porto 2. O vetor de oferta atribuído ao porto é $Q_{2p} = [0, 10000, 0, 0]$, entretanto o Navio 1 não é capaz de coletar as 10000 toneladas disponíveis do Produto 2, mesmo tendo um porão desocupado. Logo, o navio irá coletar a quantidade máxima que ele pode carregar, representado por $[0, 1800, 0, 0]$. Neste momento, como ele está lotado e não possui mais porões disponíveis, a sua rota de *pickup* está definida, conforme esquema disposto na Figura 8.

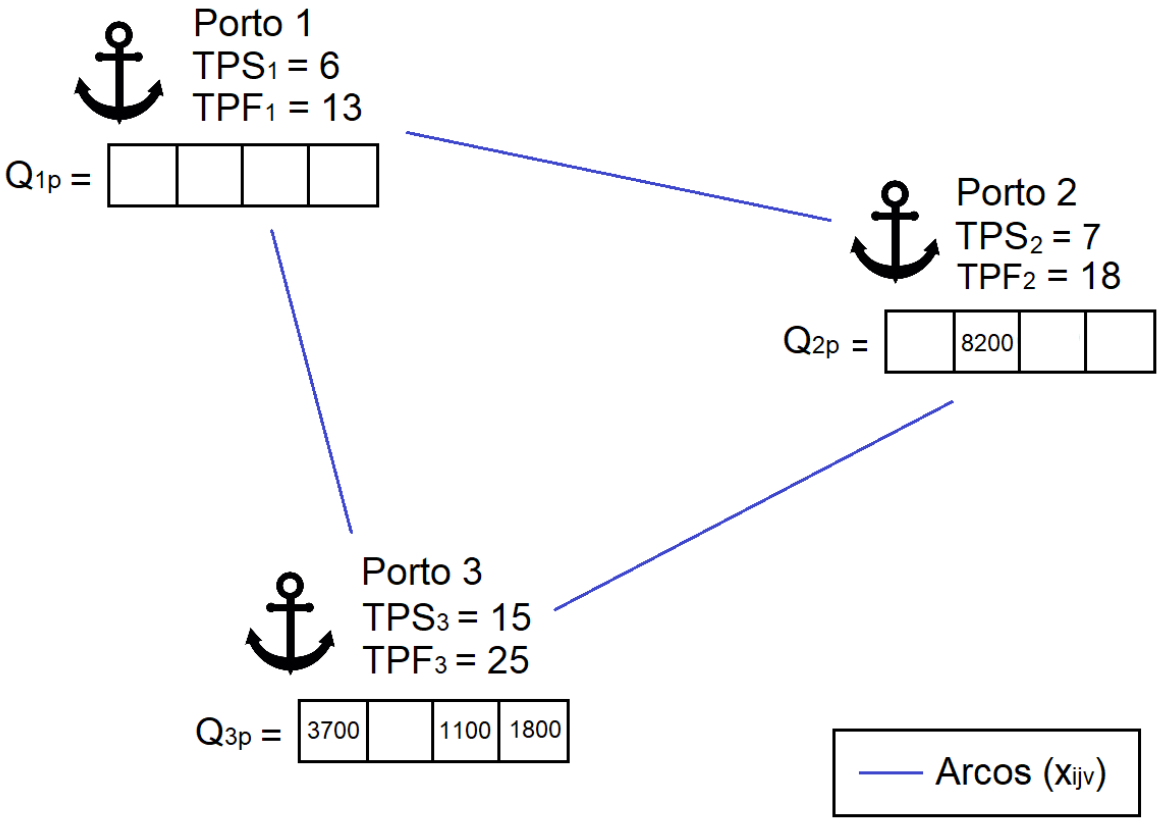
Figura 8 – Exemplo de roteamento de *pickup* para o Navio 1



Fonte: elaborado pelo autor

O vetor de produtos que está alocado para o Navio 1 é o $[3000, 1800, 5200, 0]$. Pode-se notar a ausência de produtos no Porto 1, uma vez que o Navio 1 realizou a coleta de todos. Entretanto, ainda restam produtos nos demais portos de coleta. Logo, é preciso que novos navios sejam acionados para definir outras rotas de *pickup*. A nova configuração dos produtos nos portos de coleta é apresentado na Figura 9.

Figura 9 – Exemplo de estrutura de portos de coleta após coleta do Navio 1

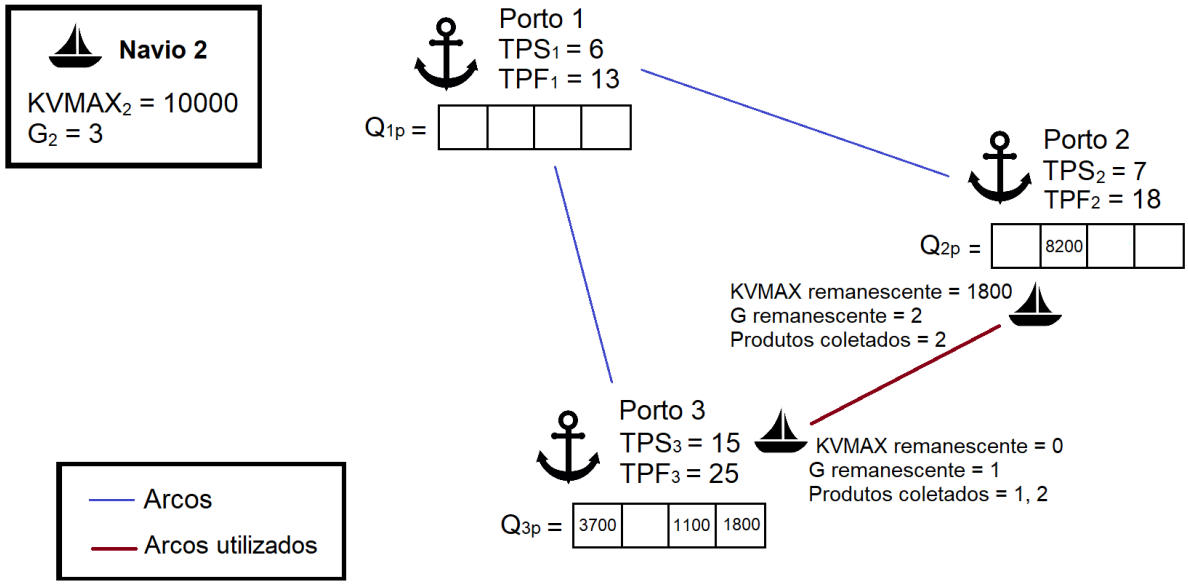


Fonte: elaborado pelo autor

O Navio 2, pelo algoritmo, deverá atracar no Porto 2, uma vez que é o porto com o menor $\frac{TPS_i+TPF_i}{2}$ e que há ainda produtos remanescentes. Ele irá coletar do porto o vetor $[0, 8200, 0, 0]$ que representa a totalidade de produtos restantes no porto. Como restam 1800 toneladas a serem preenchidas pelo navio, o próximo porto que o navio será conduzido é o Porto 3. Entretanto, devido a presença de três produtos distintos e nenhum em comum, o Navio 2 terá de fazer opções. Estas estarão sujeitas às seguintes condições:

- Seleciona-se o produto com maior quantidade no porto;
- Se for possível acomodar no navio toda a quantidade disponível no porto e ainda houver capacidade remanescente bem como porões disponíveis, carregar-se-á o navio do segundo produto com maior disponibilidade no porto, e assim sucessivamente.

Figura 10 – Exemplo de roteamento de *pickup* para o Navio 2



Fonte: elaborado pelo autor

Dessa forma, o vetor de produtos a ser acomodado no Navio 2 é o $[1800, 8200, 0, 0]$, pois o mesmo não consegue carregar em sua totalidade o produto com maior presença no Porto 3. O roteamento previsto para o Navio 2 está representado na Figura 10 enquanto a estrutura de produtos nos portos restante está representada na Figura 11. Por fim, o último navio irá atracar diretamente no Porto 3, coletando os três produtos remanescentes em $[1900, 0, 1100, 1800]$. Dessa forma, podemos resumir o trabalho realizado através da Tabela 2, onde :

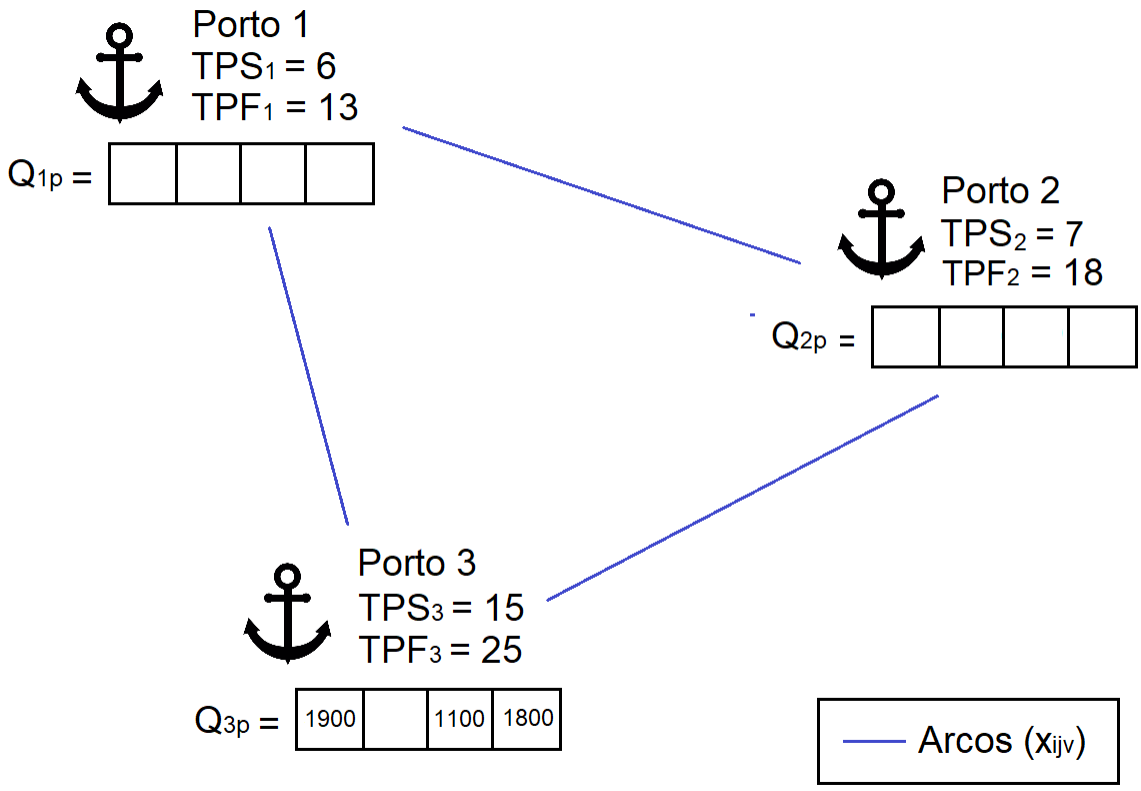
Tabela 2 – Resumo do *pickup*

Navio	Produto				(33)	(10)	(7)
	1	2	3	4			
1	3000	1800	5200	0	2	3	10000
2	1800	8200	0	0	2	2	10000
3	1900	0	1100	1800	1	3	4800

Fonte: elaborado pelo autor

As últimas colunas da Tabela 2 numeradas são referentes a algumas restrições do problema que ocorrem no *pickup* - previamente apresentadas no exemplo, bem como no Capítulo 3. Algumas restrições, entretanto, foram desprezadas para efeitos explicativos.

Figura 11 – Exemplo de estrutura de portos de coleta após coleta do Navio 1 e do Navio 2

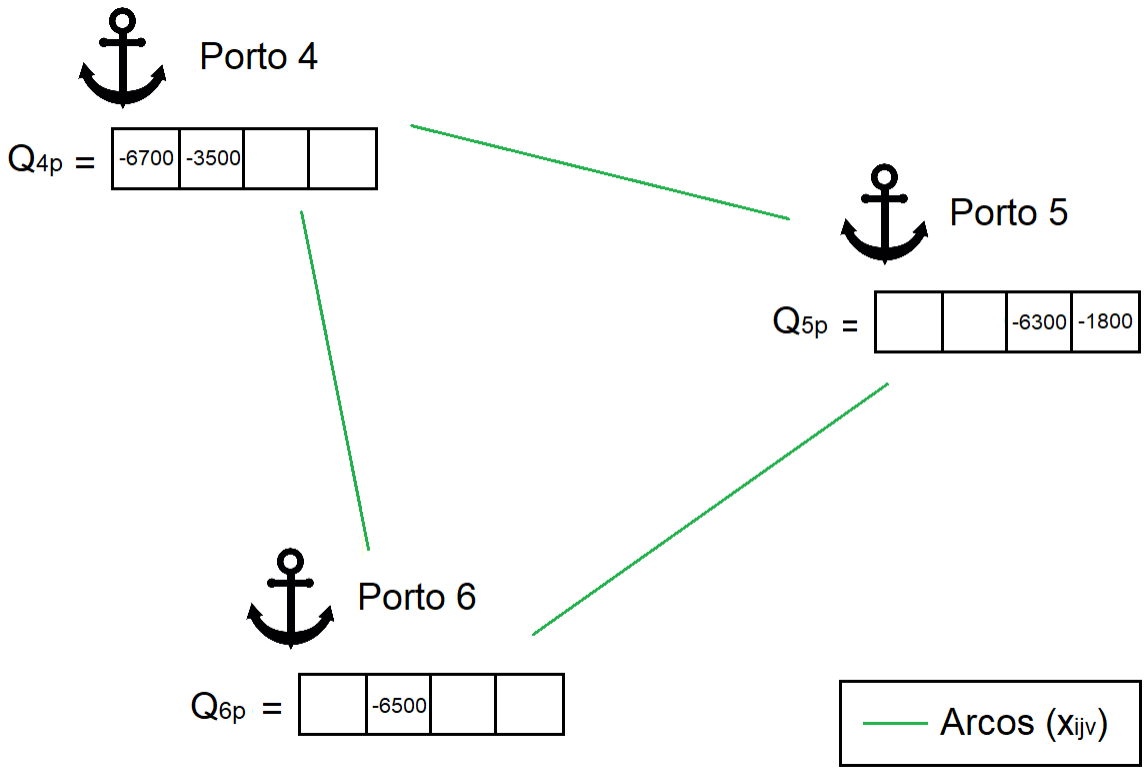


Fonte: elaborado pelo autor

4.1.1.2 *Delivery*

A fase de *delivery* funciona de forma direta. O navio designado irá realizar a entrega dos produtos no porto onde houver mais demanda pelos produtos que ele está carregando, independentemente da distância. Este valor refere-se à soma, em toneladas, da quantidade demandada, limitada pela quantidade desse produto que está sendo transportada pelo navio, expressa por $\min(l_{ijpv}, -Q_{ip})$. Se esta soma for igual para múltiplos portos, seleciona-se o com a maior diversidade de produtos. Caso o empate ainda persistir, sorteia-se um porto de entrega. Este procedimento irá ocorrer até o navio não tiver mais produtos remanescentes.

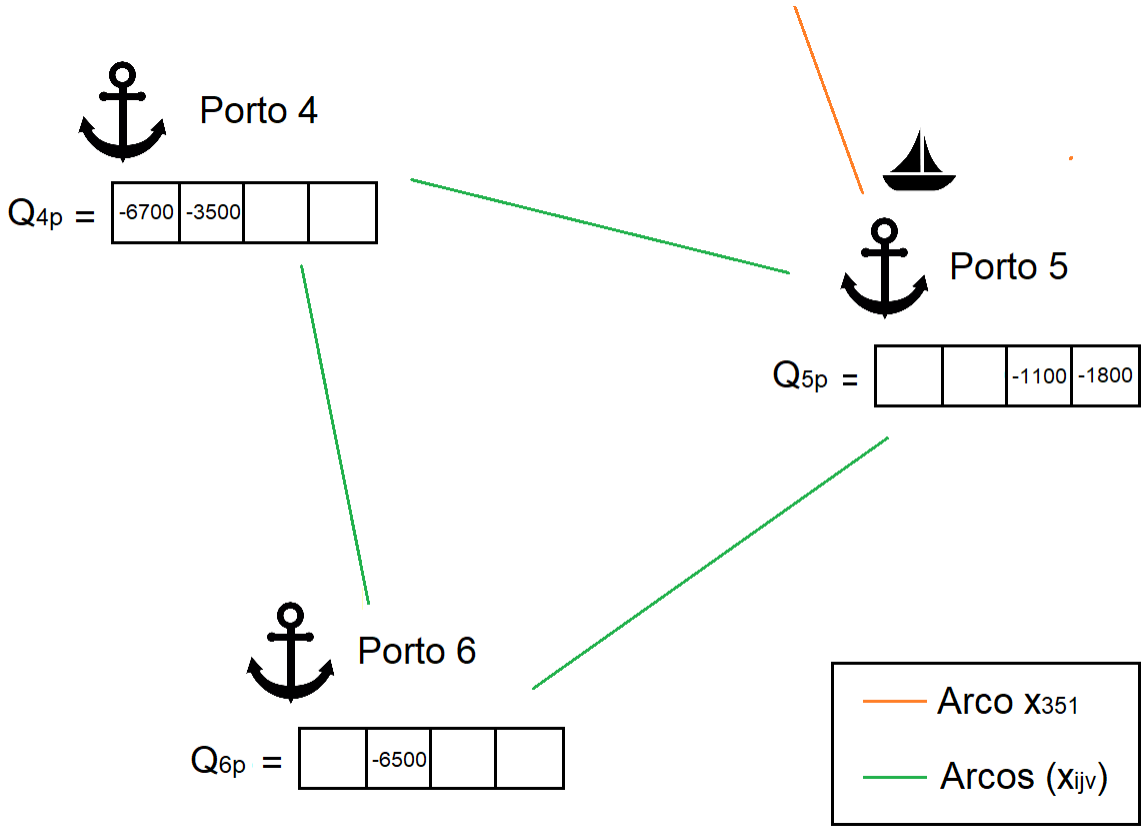
Figura 12 – Exemplo de estrutura de portos de entrega



Fonte: elaborado pelo autor

Na Figura 12, temos três portos de entrega equidistantemente dispostos. O Navio 1, que estava no Porto 2 está transportando três produtos distintos, conforme apresentado na Tabela 2. Calcula-se então as prioridades de entrega deste navio baseado na demanda dos portos e nos produtos que ele está transportando. Se o Navio 1 optar por atracar no Porto 4 inicialmente, conseguirá descarregar 2 de seus 3 produtos: $\min(3000, 6700) = 3000$, relativo ao Produto 1 e $\min(1800, 3500) = 1800$, relativo ao Produto 2, importando em um valor de 4800 toneladas. Caso o Navio 1 optar por atracar no Porto 5, conseguirá descarregar apenas um de seus 3 produtos: $\min(5200, 6300) = 5200$, relativo ao Produto 3. Por último, caso caso optar por atracar no Porto 6, conseguirá descarregar também apenas um de seus 3 produtos: $\min(1800, 6500) = 1800$. Logo, o movimento mais robusto em termos de quantidade de produtos descarregados é representado pelo arco x_{351}. A nova configuração das demandas está representada na Figura 13.

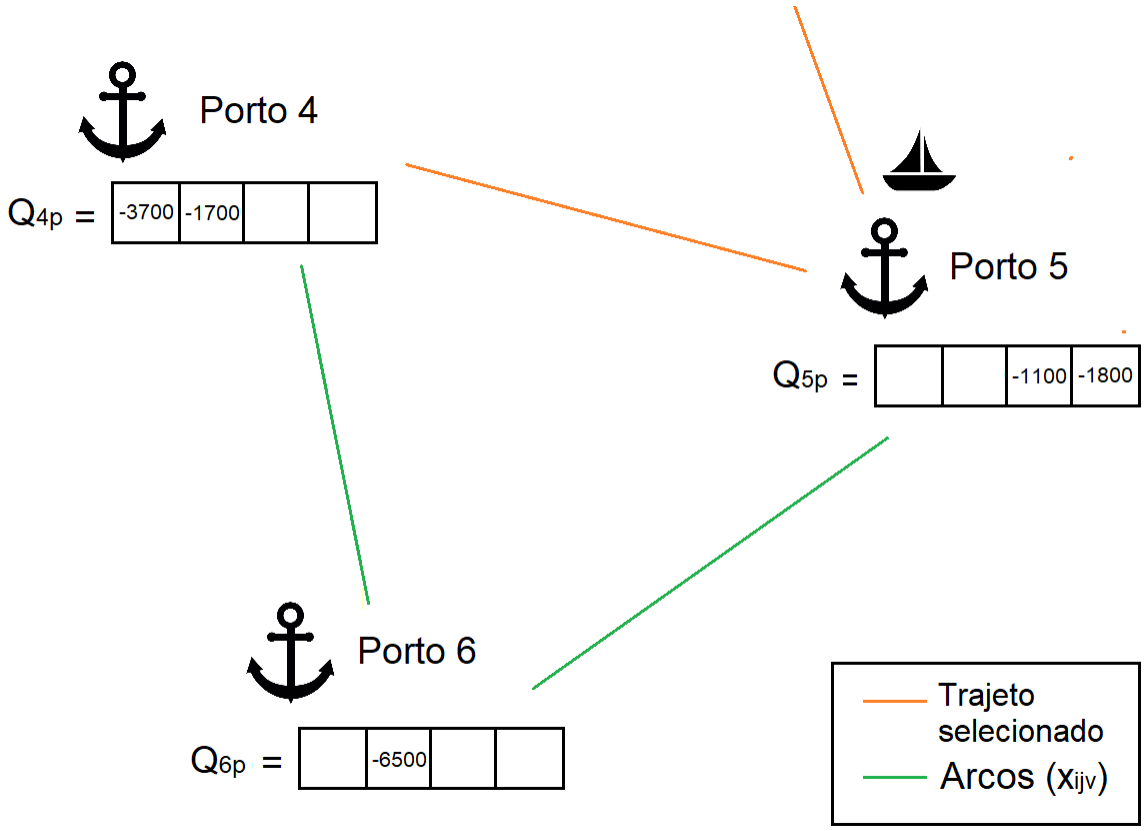
Figura 13 – Exemplo de estrutura dos portos de entrega após trabalho no arco x_{351} pelo Navio 1



Fonte: elaborado pelo autor

O Navio 1, que está no Porto 5, deverá visitar o Porto 4 ou o Porto 6 para realizar nova entrega, uma vez que ainda está carregado de produtos. Se o Navio 1 optar por atracar no Porto 4, conseguirá descarregar seus dois produtos remanescentes. Como a quantidade de produtos necessária para a entregar no porto não modificou, bem como a quantidade remanescente destes produtos no navio, mantém-se o valor de 4800 toneladas para esse movimento. Esta mesma situação se vale para o Porto 6, só que conseguirá entregar apenas 1800 toneladas. Logo, o movimento mais robusto é representado pelo arco x_{541}.

Figura 14 – Exemplo de estrutura dos portos de entrega após trabalho no arco x_{541} pelo Navio 1



Fonte: elaborado pelo autor

Dessa forma, temos que o Navio 1 está completamente descarregado e a Figura representa a nova configuração das demandas portuárias que deverão ser realizadas pelos navios sucessores.

Ademais, acrescenta-se o fato de que as soluções geradas nessa etapa não são garantidamente viáveis para o problema. Entretanto, ao se relaxar certas restrições referente à visita dos navios nos portos de coleta e entrega, essas soluções passam a ser viáveis ou estarem próximas de serem viáveis.

4.1.2 Etapa de aplicação do *relax-and-fix*

Inicialmente, todos os conjuntos de variáveis inteiras são distintamente particionados. Cada uma dessas partições é inicialmente relaxada e, em cada etapa de execução do algoritmo, uma nova partição perde a condição de relaxação e um MILP novo é resolvido. Essas partições podem ser melhor visualizadas na Tabela 3. Todos os valores das variáveis inteiras de soluções obtidas dentro de um período t_s que é atribuído a cada iteração do algoritmo são armazenadas e fixadas na etapa posterior da heurística. Acrescenta-se o fato de que nesta etapa, o objetivo primário está focado na obtenção de soluções viáveis, secundarizando a relevância da otimalidade pelo motivo de que a especificidade dessas soluções pode não levar a soluções factíveis em etapas

posteriores do algoritmo, além de demandar um maior tempo de execução descontinuadamente.

Caracteristicamente, quando $s \geq 2$, a heurística carregará na memória do programa os valores das variáveis previamente armazenados provenientes da etapa imediatamente anterior e tentará fixar como solução inicial do novo MILP. Se mais de uma solução obtida for factível, a melhor será carregada no otimizador e as demais serão utilizadas para reparação do MILP, auxiliando-o no procedimento de otimização. Entretanto, se em nenhuma das soluções parciais for verificada a viabilidade, todas elas, logo, terão esse papel auxiliar.

Para a fase de aplicação do *relax-and-fix*, foram determinadas certas ordens de particionamento dos conjuntos de variáveis, definidas como estratégias de relaxamento. Estas ordens foram elaboradas a partir do nível de importância de certos conjuntos para a resolução do problema, motivo o qual x_{ijv} é apresentado primeiramente em todas as estratégias, visto que a rede de fluxo possui extrema importância para a definição das rotas a serem percorridas.

Tabela 3 – Estratégias de relaxamento

ER	Ordem
A	$x_{ijv}, g_{pv}, y_{ijpv}, yd_{ijpv}, aux_{wv}$
B	$x_{ijv}, g_{pv}, aux_{wv}, y_{ijpv}, yd_{ijpv}$
C	$x_{ijv}, aux_{wv}, g_{pv}, y_{ijpv}, yd_{ijpv}$
D	$x_{ijv}, y_{ijpv}, g_{pv}, yd_{ijpv}, aux_{wv}$
E	$x_{ijv}, y_{ijpv}, yd_{ijpv}, g_{pv}, aux_{wv}$

Fonte: elaborado pelo autor

Conforme apresentado anteriormente, a heurística dispõe de alguns parâmetros personalizáveis, que conforme o entendimento do problema, podem ser alterados para que o algoritmo tenha uma melhor performance. Dentre esses parâmetros, destacam-se as estratégias de relaxamento (ER), a condição de paralelismo e o vetor t_s, que corresponde ao período de tempo da etapa s do algoritmo. A Tabela 8 mostra os resultados de função objetivo para 50 segundos totais de execução, em que cada etapa é responsável por $\frac{1}{5}$ desse tempo.

Tabela 4 – Resultados computacionais da Instância #2 (t_s = 10 x 5)

ER	Paralelismo	Objetivo
A	não	827910
B	não	1035060
C	não	1158640
D	não	827910
E	não	827910
A	sim	827910
B	sim	827910
C	sim	832820
D	sim	827910
E	sim	827910

Fonte: elaborado pelo autor

É valido ressaltar que, embora o vetor t_s utilizado tenha tido seus valores iguais, isso não necessariamente precisa acontecer. Em algumas instâncias mais complexas, algumas etapas podem exigir um tempo de execução superior para que ocorra, por exemplo, uma reparação bem-sucedida, ou até mesmo para popular o *pool* de soluções a fim de que a heurística consiga, em geral, melhores resultados. Logo, a estratégia de relaxamento A foi a que apresentou melhores resultados e, dessa forma, foi utilizada posteriormente para a comparação do desempenho computacional da heurística.

4.1.3 Pós-processamento

Após as etapas da fase de aplicação do *relax-and-fix* serem realizadas em sua completude, uma etapa de pós-processamento é aplicada à solução corrente no intuito de continuar o processo de melhoria dos *gaps* vigentes. Esta etapa consiste na aplicação de uma heurística de polimento presente na biblioteca do CPLEX.

O polimento de soluções pode gerar melhores soluções em situações onde boas soluções são difíceis de encontrar. Mais exigente computacionalmente do que outras heurísticas, o polimento de solução é, na verdade, uma variação do *branch-and-cut* que funciona depois que uma solução inicial está disponível. Na verdade, é necessário que pelo menos uma solução completa esteja disponível para polimento, seja uma solução produzida pelo *branch-and-cut*, ou um *MIP start* fornecido pelo usuário.

Além disso, uma etapa de reparação adicional é executada caso tenha-se obtido alguma melhora na solução corrente durante etapa de polimento. Dessa forma, utiliza-se o *pool* de soluções gerado na fase de polimento e ativa-se o procedimento de reparação. Algumas instâncias se beneficiaram positivamente, obtendo melhoras substanciais na função objetivo.

4.1.3.1 Paralelismo

O CPLEX possui um parâmetro chamado `IloCplex::Param::Parallel` que permite controlar o caminho em que a árvore de soluções do problema navegará. As opções possíveis são: oportunístico, *default* ou determinístico.

Nesse contexto, determinístico significa que várias execuções com o mesmo modelo, nas mesmas configurações de CPU e de parâmetros reproduzirão o mesmo caminho e resultados de solução. Em contraste, a abordagem oportunística implica que mesmo pequenas diferenças no tempo entre *threads* do processador ou na ordem em que as tarefas são executadas em diferentes *threads* podem conduzir a um caminho de solução diferente e consequentemente diferentes *timings* ou diferentes vetores de solução durante a otimização executada em *threads* paralelos.

Optou-se pela utilização do paralelismo no algoritmo, pois apresentou, em média, resultados melhores em comparação à abordagem *default*.

4.2 ALGORITMO

Seja $\mathcal{S}$ a representação dos conjuntos de variáveis inteiras do problema. Temos que $\mathcal{P}$ é a representação do *pool* de soluções do problema, em que cada p_s representa alguma solução do conjunto de soluções do sub-problema e p_s^n representa uma solução única desse conjunto de soluções do sub-MILP subsequente. O subconjunto de soluções de cada etapa s é definido como P_s. Considere ainda que p_s^* seja a melhor solução possível dentre os p_s obtidos no procedimento de otimização. Ainda, temos que $\mathcal{T}$ é o vetor de entrada com os limites máximos de tempo t_s que o CPLEX deve dispender em cada etapa s do algoritmo.

Em termos das bibliotecas utilizadas, define-se como `ILOINT` as variáveis inteiras do problema e `ILOFLOAT` as variáveis reais do problema. A cada nova iteração, um conjunto de variáveis têm a propriedade `ILOFLOAT` modificada para `ILOINT`, de forma que na última etapa de aplicação do *relax-and-fix* tenha-se o problema original. Além disso, o parâmetro `RepairTries` que é acionado no algoritmo para administrar o procedimento de reparação.

Algoritmo 1: $HeuristicaRelaxAndFix(EB, ET, \mathcal{T})$

Relaxa condições de integralidade em $s \in \mathcal{S}$;
para $s = 1, ..., |\mathcal{S}|$ **faça**
  se $s \geq 2$ **então**
    se $\exists\, p_{s-1} \in P_{s-1}$ tal que p_{s-1} seja viável **então**
      Carrega os valores de p_{s-1}^* utilizando a função `readMIPStarts`;
      Retira p_{s-1}^* do sub-*pool* de soluções P_{s-1};
      se $|P_{s-1}| \geq 2$ **então**
        Ativa `IloCplex::RepairTries` utilizando as demais soluções em P_{s-1};
      fim
    fim
  fim
  Adiciona condição de integralidade referente à ER utilizada (`ILOFLOAT` $\rightarrow$ `ILOINT`);
  Resolve o sub-MILP com limite de tempo t_s;
  Atualiza P_s com as soluções obtidas;
fim

4.3 PARÂMETROS

Certos parâmetros do CPLEX são ativados conjuntamente em cada etapa. Dentre eles, os parâmetros que permanecem ativados são os de geração de *cutting planes*, como:

- `IloCplex::Cliques`;
- `IloCplex::Covers`;

- `IloCplex::DisjCuts`;
- `IloCplex::FlowCovers`;
- `IloCplex::FlowPaths`;
- `IloCplex::GUBCovers`;
- `IloCplex::ImplBd`;
- `IloCplex::MIRCuts`;
- `IloCplex::MCFCuts`;
- `IloCplex::ZeroHalfCuts`.

Estes parâmetros acima são intensificados para que a busca por possíveis soluções seja acelerada. Além destes, o parâmetro `IloCplex::Param::MIP::Strategy::Probe` é ativado no seu nível máximo (3) em cada etapa do algoritmo. O *probing* é uma técnica que analisa as implicações lógicas do problema ao fixar cada variável binária em 0 ou 1 (IBM, 2016), simplificando-o muitas vezes, aumentando, dessa forma, a probabilidade de se encontrar uma solução viável para o subproblema em questão.

O parâmetro `IloCplex::Param::Emphasis::MIP` é utilizado para modificar a ênfase do mecanismo de otimização. Seu *default* é 2, de forma que o otimizador processa um "equilíbrio" entre otimalidade e viabilidade. Entretanto, no algoritmo, este valor é alterado para 1, que prioriza a viabilidade da solução em detrimento da otimalidade. Como o algoritmo é uma heurística com capacidade limitada de ver como uma decisão inicial afeta os intervalos subsequentes, pode não ser vantajoso resolver cada subproblema em sua otimalidade (UGGEN et al., 2013).

Alguns parâmetros são ativados pela heurística em momentos cruciais do algoritmo, como os referentes à reparação: `IloCplex::Param::MIP::Limits::RepairTries` e `IloCplex::Param::MIP::Limits::SubMIPNodeLim`. O primeiro modifica o número de tentativas da heurística de reparação do CPLEX utilizando os valores obtidos na etapa imediatamente anterior do algoritmo, estes conhecidos como *MIP starts*. O segundo modifica a profundidade de reparação realizada. Ambos os parâmetros aqui são intensificados, com o primeiro sendo alterado do *default* 1 para 1000 e o segundo de 500 para 10000.

Outros parâmetros são apenas acionados na fase de pós-processamento. Dentre eles, destaca-se o `IloCplex::PolishAfterIntSol`, que aciona uma heurística interna de polimento da solução corrente para tentar melhorá-la.

4.4 *MIP STARTS*

Os *MIP starts* armazenam os valores das variáveis que compõem a solução de um problema e estão codificados em XML (*Extensible Markup Language*). Estes dados são armazenados

em múltiplas entradas (cada arquivo pode conter diversos *MIP starts*) e carregados na memória do algoritmo na fase de aplicação do *relax-and-fix*. Os dados assim estão estruturados para que as funções do CPLEX consigam carregar os dados corretamente, uma vez que as bibliotecas utilizadas foram implementadas para tratar os dados dessa forma.

Figura 15 – Exemplo de *MIP start*

```
<?xml version = "1.0" encoding="UTF-8" standalone="yes"?>
<CPLEXSolutions version="1.2">
 <CPLEXSolution version="1.2">
  <header
    problemName="IloCplex"
    solutionName="m1"
    solutionIndex="0"
    MIPStartEffortLevel="4"
    writeLevel="2"/>
  <variables>
   <variable name="x[0][HER][TBN1]" index="36" value="1"/>
   <variable name="x[0][HER][TBN2]" index="37" value="0"/>
   <variable name="x[0][HER][TBN3]" index="38" value="0"/>
   <variable name="x[0][HER][TBN4]" index="39" value="0"/>
   <variable name="x[0][SLU][TBN1]" index="40" value="0"/>
   <variable name="x[0][SLU][TBN2]" index="41" value="1"/>
   <variable name="x[0][SLU][TBN3]" index="42" value="0"/>
   <variable name="x[0][SLU][TBN4]" index="43" value="0"/>
   <variable name="x[0][BRU][TBN1]" index="44" value="0"/>
   <variable name="x[0][BRU][TBN2]" index="45" value="0"/>
   <variable name="x[0][BRU][TBN3]" index="46" value="1"/>
   <variable name="x[0][BRU][TBN4]" index="47" value="0"/>
   <variable name="x[0][MUU][TBN1]" index="48" value="0"/>
   <variable name="x[0][MUU][TBN2]" index="49" value="0"/>
   <variable name="x[0][MUU][TBN3]" index="50" value="0"/>
   <variable name="x[0][MUU][TBN4]" index="51" value="0"/>
   <variable name="x[0][VDC][TBN1]" index="52" value="0"/>
   <variable name="x[0][VDC][TBN2]" index="53" value="0"/>
   <variable name="x[0][VDC][TBN3]" index="54" value="0"/>
```

Fonte: elaborado pelo autor

Enquanto certas definições são mantidas como *default*, como a codificação padrão em UTF-8, `problemName`, `solutionName`, `solutionIndex` e `writeLevel`, que apenas definem aspectos de unicidade da entrada bem como diretrizes do otimizador, outras informações são gravadas que revelam-se importantes para o CPLEX, dentre elas o *header* `MIPStartEffortLevel`, que indica que tipo de comportamento o otimizador terá ao receber o *MIP start*. Através de uma função da heurística, todos os *MIP starts* armazenados têm seu valor modificado de 1 para 4, que ativa o procedimento de reparação da entrada caso ela não seja viável para cada um dos conjuntos de valores das variáveis. Ademais, mesmo que ocorra a possibilidade de múltiplas entradas viáveis, o CPLEX irá selecionar a solução com melhor função objetivo, enquanto as demais também serão utilizadas para auxiliar na obtenção de melhores resultados.

O nó `<variables>` aninha os valores das variáveis. Cada variável é gravada no arquivo de forma similar ao visualizado no modelo matemático. Por exemplo, a variável `x[0][HER][TBN1]` representa o arco x com os índices $i = \{f\}$, $j = 2$, $v = 1$. O valor 0 no índice i é uma representação matemática para o armazém artificial f, enquanto `HER` indica a abreviatura do nome de um dos portos de coleta e, por fim, `TBN1` é o nome atribuído para o

navio $v = 1$. Essas nomenclaturas auxiliam no processo de manipulação dos dados, uma vez que essas *labels* atribuídas a cada um dos componentes dos conjuntos são sempre fixas e podem ser reutilizadas, por exemplo, nas etapas de aplicação do *relax-and-fix* sem perder a referência. O atributo `value` indica o valor que a variável assume no *MIP start* correspondente. No exemplo da Figura 15, são mostradas as variáveis de arco, que são binárias, logo poderão assumir apenas os valores 0 ou 1.

5 EXPERIMENTOS COMPUTACIONAIS

Este capítulo irá introduzir o método de teste da heurística. Através de um algoritmo desenvolvido pelo autor, uma bateria extensa de instâncias aleatórias foi gerada a fim de testar computacionalmente a heurística desenvolvida. Este algoritmo possui alguns parâmetros pré-estabelecidos, dentre eles:

- Tamanho: define o tamanho da instância gerada. Dentre as possibilidades, instâncias pequenas terão $N^P \in [2,4]$, $N^D \in [3,5]$, $P \in [5,8]$ e $V \in [3,4]$. Instâncias médias terão $N^P \in [3,5]$, $N^D \in [4,7]$, $P \in [6,0]$ e $V \in [4,5]$. Por fim, instâncias grandes terão $N^P \in [4,6]$, $N^D \in [5,10]$, $P \in [8,12]$ e $V \in [4,7]$.

- Tipo de frota: define a característica dos navios a serem utilizados para o roteamento (homogênea ou heterogênea). Instâncias com frota homogênea indicam que todos os navios possuirão as mesmas características, enquanto instâncias com frota heterogênea terão navios com características individualizadas. Essas características são os parâmetros do problema relativos à frota, como CV_v, CD_v, G_v, $KVMAX_v$, $KVMIN_v$, $KVPMIN_v$, VPL_v e VDL_v.

- Tipo dos portos: define a característica dos portos a serem utilizados para o roteamento (homogêneo ou heterogêneo). Instâncias com portos homogêneos indicam que todos os portos possuirão as mesmas características, enquanto instâncias com portos heterogêneos terão portos com características individualizadas. Essas características são os parâmetros do problema relativos aos portos, como KP_j, TQ_j, TO_j, TC_j, PVL_j e TDL_j.

- Janelas de tempo: define a característica das janelas de tempo do problema (amplas ou estritas). Instâncias com janelas de tempo amplas apresentarão uma maior flexibilidade de coleta e entrega dos produtos pelos navios, enquanto instâncias com janelas de tempo estritas terão essas possibilidades reduzidas.

- Disponibilidade de produtos: define a característica da disponibilidade dos produtos nos portos de coleta e entrega (esparsa ou densa). Instâncias com disponibilidade de produtos esparsa terão uma matriz Q_{ip} com um maior número de zeros, relativo à maior concentração das requisições de coleta e entrega estarem concentradas em menos portos. Por outro lado, instâncias com disponibilidade de produtos densa terão uma matriz Q_{ip} com um menor número de zeros, ou seja, a oferta e a demanda de produtos espalhadas em mais portos. Por padrão, uma matriz esparsa tem, inicialmente, uma probabilidade de 0,2 de haver alguma oferta ou demanda do produto p no porto i, enquanto uma matriz densa tem probabilidade 0,4.

A geração dos parâmetros segue distribuição uniforme de probabilidade, variando apenas seu suporte e seus parâmetros a e b. O intervalo de possibilidades para geração dos parâmetros

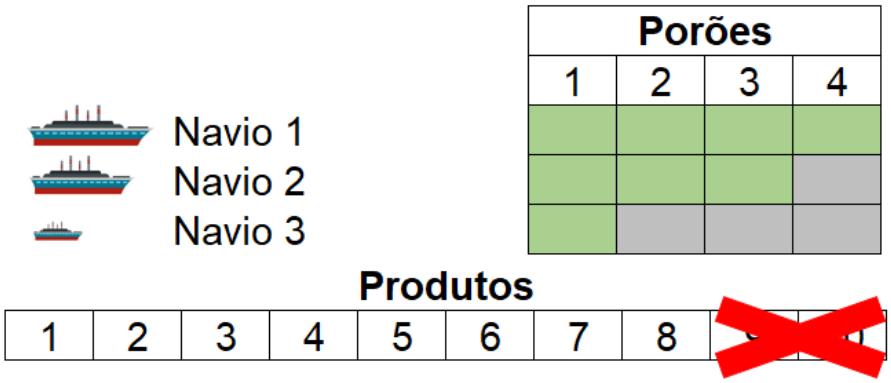
das instâncias estão intrinsecamente relacionados com os parâmetros provenientes das instâncias reais, obtidas através do trabalho de Kretschmann (2018).

5.1 DIRETRIZES PARA GERAÇÃO DE INSTÂNCIAS ALEATÓRIAS

Para que as instâncias aleatórias tenham maior probabilidade de serem viáveis, certas diretrizes foram adotadas no gerador. Dentre elas, ocorre primeiramente a definição dos parâmetros mais básicos da instância, como, por exemplo, as especificações técnicas dos navios e dos portos. Certas definições lógicas devem ser consideradas, como:

- Capacidade máxima de carga $\times$ oferta/demanda de produtos nos portos: uma instância aleatória com capacidade máxima de carga $\sum_{i \in V} KVMAX_v$ distribuída entre os navios poderá teoricamente transportar, no máximo, uma quantidade total equivalente de produtos.
- Número de produtos distintos $\times$ quantidade de porões disponíveis por navio: uma instância aleatória com uma quantidade $\sum_{i \in V} G_v$ de porões disponíveis pelos navios poderá ter, no máximo, uma quantidade equivalente de produtos distintos a serem transportados.
- Prazos de entrega $\times$ tolerância de atraso ou adiantamento: uma instância aleatória precisará de TDA_j e TDL_j diretamente relacionados aos parâmetros de tempo para que haja maior probabilidade do navio atracar nos seus destinos em tempo adequado.

Figura 16 – Exemplo da relação entre o número de produtos distintos $\times$ quantidade de porões disponíveis por navio



Fonte: elaborado pelo autor

5.1.1 Geração da matriz Q_{ip}

O procedimento de geração da matriz Q_{ip}, referente à estrutura de oferta e demanda da instância é realizada em duas etapas. Define-se a quantidade máxima de produtos (em toneladas) que serão distribuídas nos portos, bem como a quantidade limite de produtos a serem gerados, pautada pela relação apresentada anteriormente.

Figura 17 – Probabilidade de distribuição de produtos nos portos de coleta

Porto/Produto	1	2	3	4	5	6	7	8	9
C1	p	p	p	p	p	p	p	p	p
C2	p	p	p	p	p	p	p	p	p
C3	p	p	p	p	p	p	p	p	p
C4	p	p	p	p	p	p	p	p	p

Fonte: elaborado pelo autor

Primeiramente, conforme melhor visualizado na Figura 17, as probabilidades de sorteio são equivalentes para todos os portos e produtos. Uma vez que um produto p em um porto i é sorteado, uma quantidade Q_{ip} será gerada e atribuída na matriz. Além disso, essa quantidade será descontada do montante total. Entretanto, após o procedimento, certos problemas podem ocorrer, exemplificado melhor na Figura 18.

Figura 18 – Esquema indesejado de distribuição de produtos nos portos de coleta

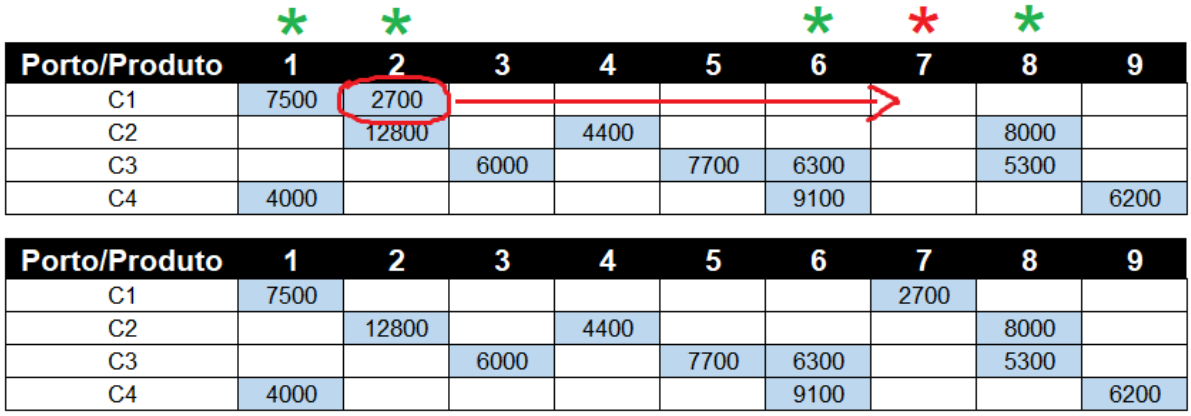
Porto/Produto	1	2	3	4	5	6	7	8	9
C1	7500	2700							
C2		12800		4400				8000	
C3			6000		7700	6300		5300	
C4	4000					9100			6200

A geração aleatória não contemplou o Produto 7

Fonte: elaborado pelo autor

No exemplo, o Produto 7 ficou sem nenhuma atribuição. Similarmente, embora com menor probabilidade de ocorrência, algum porto poderia também ficar sem nenhuma atribuição. Devido à probabilidade de atribuição, é possível que essa ou outras combinações indesejadas ocorram, e assim uma correção deve ser aplicada para que a distribuição ocorra corretamente.

Figura 19 – Redistribuição de produtos nos portos de coleta



Porto/Produto	1	2	3	4	5	6	7	8	9
C1	7500	2700							
C2		12800		4400				8000	
C3			6000		7700	6300		5300	
C4	4000					9100			6200

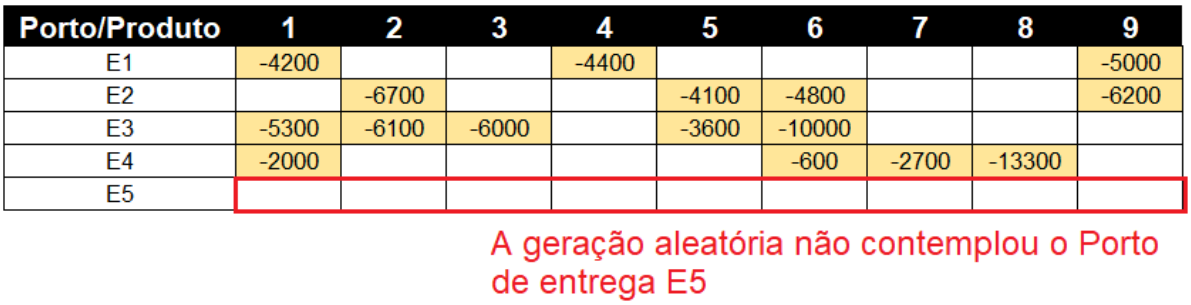
Porto/Produto	1	2	3	4	5	6	7	8	9
C1	7500						2700		
C2		12800		4400				8000	
C3			6000		7700	6300		5300	
C4	4000					9100			6200

Fonte: elaborado pelo autor

Na Figura 19, um remanejo é executado para que o Produto 7 seja contemplado na instância. Para isso, define-se os portos e/ou produtos que estão sem atribuição e os portos e/ou produtos que estão abundantes, ou seja, que têm a possibilidade de redistribuir. Entretanto, em alguns casos poderá ser impossível realizar essa redistribuição, e então a matriz Q_{ip} será repopulada.

Caso o procedimento anterior seja bem-sucedido, uma nova etapa então é atribuída no processo: a geração das demandas nos portos de entrega. Essa etapa ocorre de forma similar à anterior, verticalmente, com a principal diferença de que para cada p a quantidade máxima de produto atribuída a algum porto será $\sum_{i \in N^P} Q_{ip}$. Entretanto, ainda é possível que algum dos portos de descarga fiquem sem demanda, conforme melhor visualizado na Figura 20.

Figura 20 – Esquema indesejado de distribuição de produtos nos portos de entrega



Porto/Produto	1	2	3	4	5	6	7	8	9
E1	-4200			-4400					-5000
E2		-6700			-4100	-4800			-6200
E3	-5300	-6100	-6000		-3600	-10000			
E4	-2000					-600	-2700	-13300	
E5									

A geração aleatória não contemplou o Porto de entrega E5

Fonte: elaborado pelo autor

Todavia, a resolução dessa inconsistência é trivial, uma vez que basta deslocar a entrega de algum porto autossuficiente para o porto sem demanda, tendo apenas que respeitar a coluna referente ao produto. Melhor exemplificado na Figura 21.

Figura 21 – Redistribuição de produtos nos portos de entrega

Porto/Produto	1	2	3	4	5	6	7	8	9
E1	-4200			-4400					-5000
E2		-6700			-4100	-4800			-6200
E3	-5300	-6100	-6000		-3600	-10000			
E4	-2000					-600	-2700	-13300	
E5									

Fonte: elaborado pelo autor

5.1.2 Geração dos parâmetros de tempo

Com o mesmo objetivo das demais etapas, a geração dos parâmetros de tempo também precisaram atender certos cuidados. Foram definidos *bounds* relativos às possíveis chegadas dos navios nos portos. Supondo que um navio v irá coletar um produto p em um porto i. Temos que esse navio deverá realizá-la no intervalo definido por TPS_i e TPF_i, onde o cenário mais antecipado seria no início da janela (TPS_i) e o cenário mais atrasado seria no fim da janela (TPF_i). Considerando que esse navio v está com a capacidade máxima atingida, deverá atracar algum porto de entrega j em tempo TS_{ij}. O produto p terá um data solicitada de entrega nesse porto, representada por TD_{jp}. Ademais, outros tempos operacionais são adicionados nessa equação: TO_j, TC_j e TQ_j.

Dessa forma, pode-se gerar melhores valores para esses parâmetros de tempo, como o de TDA_j e TDL_j. Uma estimativa razoável para o TDA_j e para o TDL_j pode ser definida como $TPS_i + TS_{ij} + TO_j + TC_j + TQ_j$, considerando uma rota que parte do porto de coleta i e chega no porto de entrega j. Essa estimativa é utilizada para a geração dos parâmetros, a partir de uma distribuição uniforme com $a = TPS_i + TS_{ij} + TO_j + TC_j + TQ_j - 10$ e $b = TPS_i + TS_{ij} + TO_j + TC_j + TQ_j + 10$.

5.2 RESULTADOS

Todos os resultados apresentados nesta seção foram realizados num computador AMD Ryzen 5 2600 @ 3.40 GhZ com 16GB de RAM. Os cálculos foram realizados através da heurística desenvolvida em C++ para o *relax-and-fix* e adicionalmente de uma heurística construtiva desenvolvida em Python. A Tabela 6 apresenta os resultados computacionais obtidos através do *relax-and-fix* (RaF) e do *relax-and-fix* acoplado da heurística construtiva (RaF + C) comparados com os resultados obtidos pela programação em C++ do CPLEX.

Algumas características das instâncias geradas estão dispostas abaixo, na Tabela 5. As instâncias do grupo A foram construídas pelo gerador, enquanto as instâncias do grupo N foram desenvolvidas pelo autor. As instâncias aleatórias foram geradas com variações nos parâmetros de entrada explicitadas no capítulo anterior, a fim de aumentar a variabilidade e a possibilidade

de análises posteriores.

Tabela 5 – Cardinalidade dos conjuntos das instâncias aleatórias

Instância	$\|N^P\|$	$\|N^D\|$	$\|P\|$	$\|V\|$
#A1	2	4	7	4
#A2	4	5	6	3
#A3	3	3	8	4
#A4	2	5	6	4
#A5	4	5	7	3
#A6	2	5	6	3
#A7	3	5	7	4
#A8	4	3	6	4
#A9	4	6	7	4
#A10	5	7	6	5
#A11	4	4	6	5
#A12	5	4	7	5
#A13	5	4	6	5
#A14	4	4	6	4
#N1	3	6	8	4
#N2	5	5	7	4
#N3	5	5	8	4
#N4	5	5	7	5
#N5	5	7	9	4
#N6	5	8	10	4
#N7	6	9	10	4

Fonte: elaborado pelo autor

Em geral, as instâncias fictícias apresentam características específicas que as tornam mais complexas de serem resolvidas, mesmo as com número reduzido de portos e produtos. Isso ocorre devido ao fato de que os parâmetros gerados não seguem algum tipo de planejamento, o que ocorre no processo decisório da empresa, e dessa forma, a disposição de produtos nos portos e as janelas de tempo podem fazer com que se tenha um conjunto de soluções mais difícil de ser encontrado pelo algoritmo.

Os resultados estão dispostos na Tabela 6, com a função objetivo e o tempo de CPU, bem como o *gap* da heurística em relação ao valor do CPLEX. Foi fixado o tempo de execução do CPLEX num máximo de 7200 segundos.

Tabela 6 – Resultados computacionais das instâncias aleatórias

Instância	CPLEX		RaF			RaF + C		
	F.O.	Tempo (s)	F.O.	Tempo (s)	Gap (%)	F.O.	Tempo (s)	Gap (%)
#A1	1206770	266	1206770	100	0,00	1206770	100	0,00
#A2	2177327	7200	2105837	900	-3,28	2158489	900	-0,87
#A3	1279264	100	1279264	50	0,00	1279264	50	0,00
#A4	1575446	7200	1575446	900	0,00	1575446	900	0,00
#A5	3986998	7200	3617332	2500	-9,27	3591041	2500	-9,93
#A6	1890101	1363	1890101	150	0,00	1890101	150	0,00
#A7	1912038	7200	1912038	900	0,00	1912038	900	0,00
#A8	5299088	7200	2780194	1500	-47,53	3051703	1500	-42,41
#A9	4221534	7200	3578946	1500	-15,22	3621474	1500	-8,56
#A10	2361894	7200	2215006	1500	-6,22	2231430	1500	-5,52
#A11	1930020	7200	2022903	1500	4,81	1930020	1500	0,00
#A12	2315048	7200	2224670	1500	-3,90	2224670	1500	-3,90
#A13	1542586	7200	1542586	900	0,00	1542586	900	0,00
#A14	4328823	7200	2288063	1500	-47,14	2310964	1500	-46,61
#N1	3092020	7200	2391940	1500	-22,64	2391940	1500	-22,64
#N2	2155400	7200	2159870	1500	0,21	2155400	1500	0,00
#N3	5031280	7200	4085690	1500	-18,79	3710430	1500	-26,25
#N4	3250960	7200	2726400	1500	-16,14	2810900	1500	-13,54
#N5	2295680	7200	1964790	1500	-14,41	2010700	1500	-12,41
#N6	-	7200	2439760	2500	-	2439760	2500	-
#N7	-	7200	5880110	3500	-	5880110	3500	-

Em termos de resultados, percebemos uma substancial diferença em termos de função objetivo, e conforme o aumento da complexidade do problema, a diferença entre os resultados do CPLEX e da heurística tendem a ser mais pronunciadas. A diferença também se dá em relação ao tempo de execução, em que a heurística performa muito mais rapidamente, além de trazer os melhores resultados. Para as instâncias selecionadas, o *gap* médio do *relax-and-fix* para o CPLEX foi de -10,50% e do *relax-and-fix* com a construtiva foi de -9,14%. Se tratando de uma vantagem econômica, pode-se considerar bastante relevante uma redução de aproximadamente 10% no valor total do roteamento.

Com o aumento do tamanho das instâncias, o CPLEX começou a apresentar dificuldades para obtenção de soluções. Para as instâncias N6 e N7 não foi possível encontrar uma solução viável no tempo determinado de 7200 segundos, enquanto o *relax-and-fix* apresentou soluções em menos de uma hora.

Devido ao caráter $\mathcal{NP}$-difícil do problema, o aumento de cardinalidade de qualquer conjunto implica, em média, num incremento exponencial do tempo necessário de execução da heurística. Enquanto para instâncias pequenas muitas vezes obtém-se solução ótima em segundos, a adição de um único porto de coleta, por exemplo, nessa instância, munido das adaptações paramétricas necessárias, poderá causar um aumento do tempo para a escala dos minutos.

6 AVALIAÇÃO DA HEURÍSTICA

A avaliação da heurística também foi balizada através de instâncias reais obtidas através da empresa de fertilizantes químicos que norteou este trabalho. Analogamente ao desenvolvido no capítulo anterior, este apresentará resultados através da comparação com o método corrente, utilizado na empresa para se obter soluções, entretanto sem a utilização de programação linear inteira. As instâncias reais utilizadas tiveram algumas características específicas, contendo um número intermediário de portos de coleta ($|N^P| \leq 4$), portos de descarga ($|N^D| \leq 5$), produtos ($|P| \leq 8$) e navios ($|V| \leq 4$). Comparativamente às instâncias aleatórias, essas tiveram um grau de complexidade menor.

Os resultados obtidos pela heurística foram comparados com os resultados obtidos pela empresa e por Kretschmann (2018).

Tabela 7 – Cardinalidade dos conjuntos das instâncias reais

Instância	$\|N^P\|$	$\|N^D\|$	$\|P\|$	$\|V\|$
#1	2	5	5	3
#2	2	5	6	3
#3	3	5	6	3
#4	3	5	7	3
#5	4	5	8	4

Fonte: elaborado pelo autor

Na Tabela 6 temos algumas características das instâncias que terão os resultados discriminados a seguir.

Tabela 8 – Comparação das funções objetivo de cada método

Instância	Função objetivo (R$)				Gap (%)
	Empresa	CPLEX	RaF	RaF + C	
#1	1513310	1035600	1035600	1035600	0.00
#2	2086000	827910	827910	827910	0.00
#3	1075550	998740	998740	998740	0.00
#4	1701770	1207590	1206560	1206560	-0.08
#5	-	1842890	1688320	1687420	∞

Fonte: elaborado pelo autor

A análise dos resultados nos mostra que, para as instâncias menores, pouca diferença emergiu entre o *relax-and-fix* puro e sua versão com a utilização da heurística construtiva adicional. Além disso, comparativamente aos resultados obtidos por Kretschmann (2018), as diferenças não foram exatamente significativas em termos de função objetivo. Entretanto, para a instância mais robusta (#5), as diferenças entre a heurística e os demais métodos começaram a ser mais visíveis: tanto a empresa quanto Kretschmann (2018) não obtiveram soluções para

a instância em tempo hábil, enquanto a heurística trouxe resultados de qualidade em poucos minutos de execução.

Todavia, neste capítulo, uma comparação de maior interesse se dá nos tempos de execução: tanto do procedimento sem utilização de MILP quanto da utilização do IBM CPLEX Optimization Studio com a heurística. Essas diferenças se mostraram pronunciadas e podem ser observadas na Tabela 9.

Tabela 9 – Comparação dos tempos de execução de cada método

Instância	Tempo (s)			Δt (%)
	Kretschmann (2018)	RaF	RaF + C	
#1	192136	70	70	-99.96
#2	7289	35	35	-99.52
#3	3409	7	7	-99.79
#4	>200000	70	70	N/A
#5	-	900	900	∞

Fonte: elaborado pelo autor

Todas as instâncias tiveram uma economia de tempo superior a 99% em relação ao trabalho de Kretschmann (2018). Porém, enquanto mais simples de modelar, o Optimization Studio, por ser uma linguagem interpretada, é mais lento ao se comparar com um programa desenvolvido em uma linguagem de programação compilada, como é o caso o C++, utilizado para a implementação da heurística. Logo, uma comparação mais interessante foi aplicada ao traduzir o modelo para o C++ utilizando as bibliotecas disponíveis do CPLEX, como já realizado no capítulo anterior com as instâncias aleatórias. Os resultados estão dispostos na Tabela 10.

Tabela 10 – Resultados das instâncias reais obtidos no CPLEX em C++

Instância	Função objetivo	Tempo (s)	Gap (%)	Δt (%)
#1	1035600	848	0.00	-91.75
#2	827910	84	0.00	-58.33
#3	998740	138	0.00	-94.93
#4	1206560	2490	0.00	-97.19
#5	1842890	7200	-8.44	-87.50

Fonte: elaborado pelo autor

Para os resultados obtidos no CPLEX, o limite de tempo determinado foi de 7200 segundos, o mesmo fixado para as instâncias aleatórias. Os *gaps* da tabela referem-se a diferença em percentual da função objetivo e do tempo obtido na heurística em relação ao obtido no CPLEX. Nota-se que mesmo com os resultados mais competitivos fornecidos pelo otimizador, a heurística superou-o em tempo para todas as instâncias, e na mais difícil (#5), a qual o CPLEX não atingiu à otimalidade no tempo pré-definido, o *relax-and-fix* obteve um resultado expressivamente melhor com um tempo de execução, em média, 8 vezes menor.

7 CONCLUSÃO

A heurística *relax-and-fix* demonstrou ser bastante apropriada para resolver o problema marítimo apresentado, uma vez que superou os resultados do otimizador tradicional em quase todas as instâncias testadas. Além disso, a qualidade das soluções obtidas em tempo reduzido é um motivador para seu uso no mundo real.

O desenvolvimento de um gerador de instâncias aleatórias também contribuiu de maneira relevante para os resultados e ajustes finos da heurística, além de incorporar à literatura um conjunto de instâncias com *benchmark* para que estudos posteriores possam se valer destas estudadas neste trabalho.

Limitações do estudo estão situadas na escalabilidade da heurística, ou seja, no limite teórico do tamanho das instâncias que podem ser resolvidas em tempo hábil pelo programa. Nota-se grande evolução nesse aspecto em relação ao trabalho de Kretschmann (2018), entretanto, ausenta-se outros tipos de comparações, uma vez que o problema apresentado é bastante específico. Mesmo assim, o escopo do mundo real foi atendido de forma plena, o que motiva a exploração de novos métodos para aprimorar e complementar este trabalho.

Finalmente, acredita-se que o trabalho incorporou de forma satisfatória os conhecimentos exprimidos no trabalho de Kretschmann (2018) para adaptar-se a uma diferente realidade de aplicação e desenvolvimento da heurística *relax-and-fix*.

REFERÊNCIAS

AGARWAL, R.; ERGUN, Ö. Ship scheduling and network design for cargo routing in liner shipping. **Transportation Science**, INFORMS, v. 42, n. 2, p. 175–196, 2008.

AL-KHAYYAL, F.; HWANG, S.-J. Inventory constrained maritime routing and scheduling for multi-commodity liquid bulk, part i: Applications and model. **European Journal of Operational Research**, Elsevier, v. 176, n. 1, p. 106–130, 2007.

APPELGREN, L. H. A column generation algorithm for a ship scheduling problem. **Transportation Science**, INFORMS, v. 3, n. 1, p. 53–68, 1969.

ARNESEN, M. J.; GJESTVANG, M.; WANG, X.; FAGERHOLT, K.; THUN, K.; RAKKE, J. G. A traveling salesman problem with pickups and deliveries, time windows and draft limits: Case study from chemical shipping. **Computers & Operations Research**, Elsevier, v. 77, p. 20–31, 2017.

AYTEKIN, M. **Applying a Fix-and-Relax Heuristics to US Navy Force Structure Planning**. [S.l.], 2002.

BARNHART, C.; LAPORTE, G. **Handbooks in operations research and management science: transportation**. [S.l.]: Elsevier, 2006. v. 14.

BATTARRA, M.; PESSOA, A. A.; SUBRAMANIAN, A.; UCHOA, E. Exact algorithms for the traveling salesman problem with draft limits. **European Journal of Operational Research**, Elsevier, v. 235, n. 1, p. 115–128, 2014.

BERALDI, P.; GHIANI, G.; GRIECO, A.; GUERRIERO, E. Fix and relax heuristic for a stochastic lot-sizing problem. **Computational Optimization and Applications**, Springer, v. 33, n. 2-3, p. 303–318, 2006.

BERALDI, P.; GHIANI, G.; GRIECO, A.; GUERRIERO, E. Rolling-horizon and fix-and-relax heuristics for the parallel machine lot-sizing and scheduling problem with sequence-dependent set-up costs. **Computers & Operations Research**, Elsevier, v. 35, n. 11, p. 3644–3656, 2008.

BRØNMO, G.; CHRISTIANSEN, M.; NYGREEN, B. Ship routing and scheduling with flexible cargo sizes. **Journal of the Operational Research Society**, Taylor & Francis, v. 58, n. 9, p. 1167–1177, 2007.

CHRISTIANSEN, M. Decomposition of a combined inventory and time constrained ship routing problem. **Transportation science**, INFORMS, v. 33, n. 1, p. 3–16, 1999.

CHRISTIANSEN, M.; FAGERHOLT, K.; NYGREEN, B.; RONEN, D. Ship routing and scheduling in the new millennium. **European Journal of Operational Research**, Elsevier, v. 228, n. 3, p. 467–483, 2013.

DANTZIG, G.; RAMSER, J. *The Truck Dispatching Problem*. **INFORMS**, out. 1959.

DILLENBERGER, C.; ESCUDERO, L. F.; WOLLENSAK, A.; ZHANG, W. On practical resource allocation for production planning and scheduling with period overlapping setups. **European Journal of Operational Research**, Elsevier, v. 75, n. 2, p. 275–286, 1994.

ESCUDERO, L. F.; SALMERON, J. On a fix-and-relax framework for a class of project scheduling problems. **Annals of Operations Research**, Springer, v. 140, n. 1, p. 163, 2005.

FERREIRA, D.; MORABITO, R.; RANGEL, S. Relax and fix heuristics to solve one-stage one-machine lot-scheduling models for small-scale soft drink plants. **Computers & Operations Research**, Elsevier, v. 37, n. 4, p. 684–691, 2010.

FRISKE, M. W.; BURIOL, L. S. Applying a relax-and-fix approach to a fixed charge network flow model of a maritime inventory routing problem. In: SPRINGER. **International Conference on Computer Languages**. [S.l.], 2018. p. 3–16.

GAREY, M. R.; JOHNSON, D. S. **Computers and intractability**. [S.l.]: wh freeman New York, 2002. v. 29.

IBM. **Probing**. 2016. Urlhttps://www.ibm.com/support/knowledgecenter/SSSA5P$_1$2.7.0/$ilog.odms.cplex.help/CPLEX/UsrM$

KELLY, J. D.; MANN, J. L. Flowsheet decomposition heuristic for scheduling: a relax-and-fix method. **Computers & chemical engineering**, Elsevier, v. 28, n. 11, p. 2193–2200, 2004.

KORSVIK, J. E.; FAGERHOLT, K. A tabu search heuristic for ship routing and scheduling with flexible cargo quantities. **Journal of Heuristics**, Springer, v. 16, n. 2, p. 117–137, 2010.

KRETSCHMANN, E. **Uma abordagem ao Vehicle Routing Problem with Pickup and Delivery and Time Windows em navios: um caso de uma indústria química**. Dissertação (Mestrado), Porto Alegre, Brazil, jul. 2018.

LI, D.; HUANG, H.-C.; CHEW, E.-P.; MORTON, A. Simultaneous fleet assignment and cargo routing using benders decomposition. In: **Container Terminals and Cargo Systems**. [S.l.]: Springer, 2007. p. 315–331.

MALAGUTI, E.; MARTELLO, S.; SANTINI, A. The traveling salesman problem with pickups, deliveries, and draft limits. **Omega**, Elsevier, v. 74, p. 50–58, 2018.

MOHAMMADI, M.; GHOMI, S. F.; KARIMI, B.; TORABI, S. A. Rolling-horizon and fix-and-relax heuristics for the multi-product multi-level capacitated lotsizing problem with sequence-dependent setups. **Journal of Intelligent Manufacturing**, Springer, v. 21, n. 4, p. 501–510, 2010.

PAPAGEORGIOU, D. J.; NEMHAUSER, G. L.; SOKOL, J.; CHEON, M.-S.; KEHA, A. B. Mirplib–a library of maritime inventory routing problem instances: Survey, core model, and benchmark results. **European Journal of Operational Research**, Elsevier, v. 235, n. 2, p. 350–366, 2014.

RAKKE, J. G.; CHRISTIANSEN, M.; FAGERHOLT, K.; LAPORTE, G. The traveling salesman problem with draft limits. **Computers & Operations Research**, Elsevier, v. 39, n. 9, p. 2161–2167, 2012.

RONEN, D. Cargo ships routing and scheduling: Survey of models and problems. **European Journal of Operational Research**, Elsevier, v. 12, n. 2, p. 119–126, 1983.

SONG, D.-P.; DONG, J.-X. Cargo routing and empty container repositioning in multiple shipping service routes. **Transportation Research Part B: Methodological**, Elsevier, v. 46, n. 10, p. 1556–1575, 2012.

TANIGUCHI, E.; YAMADA, T.; TAMAISHI, M.; NORITAKE, M. Effects of designated time on pickup/delivery truck routing and scheduling. **WIT Transactions on The Built Environment**, WIT Press, v. 36, 1970.

TODOSIJEVIĆ, R.; MJIRDA, A.; MLADENOVIĆ, M.; HANAFI, S.; GENDRON, B. A general variable neighborhood search variants for the travelling salesman problem with draft limits. **Optimization Letters**, Springer, v. 11, n. 6, p. 1047–1056, 2017.

UGGEN, K. T.; FODSTAD, M.; NØRSTEBØ, V. S. Using and extending fix-and-relax to solve maritime inventory routing problems. **Top**, Springer, v. 21, n. 2, p. 355–377, 2013.

UNCTAD, G. World investment report: Towards a new generation of investment policies. **Unite Nations. New York and Geneva**, 2012.

UNCTAD, W. United nations conference on trade and development. **Review of Maritime Transport**, 2014.