UM ALGORITMO DE BUSCA HÍBRIDO PARA O PROBLEMA DE ROTEAMENTO DE EMBARCAÇÕES DE SUPRIMENTO PERIÓDICO

Juliana Beatriz Carvalho de Oliveira Soares Boucher

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Produção, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Produção.

Orientadores: Virgílio José Martins Ferreira Filho
Laura Silvia Bahiense da Silva Leite

Rio de Janeiro
Maio de 2018

UM ALGORITMO DE BUSCA HÍBRIDO PARA O PROBLEMA DE ROTEAMENTO DE EMBARCAÇÕES DE SUPRIMENTO PERIÓDICO

Juliana Beatriz Carvalho de Oliveira Soares Boucher

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE PRODUÇÃO.

Examinada por:

Prof. Vírgilio José Martins Ferreira Filho, D.Sc.

Prof. Laura Silvia Bahiense da Silva Leite, D.Sc.

Prof. Glaydston Mattos Ribeiro, D.Sc.

Prof. Luiz Satoru Ochi, D.Sc.

Prof. Edilson Fernandes Arruda, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
MAIO DE 2018

Boucher, Juliana Beatriz Carvalho de Oliveira Soares

Um Algoritmo de Busca Híbrido para o Problema de Roteamento de Embarcações de Suprimento Periódico /Juliana Beatriz Carvalho de Oliveira Soares Boucher. – Rio de Janeiro: UFRJ/COPPE, 2018.

XIV, 92 p.: il.; 29, 7cm.

Orientadores: Virgílio José Martins Ferreira Filho
Laura Silvia Bahiense da Silva Leite

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia de Produção, 2018.

Referências Bibliográficas: p. 86 – 92.

1. periodic vehicle routing problem. 2. offshore logistics. 3. adaptive large neighborhood search. 4. clustering search. I. Ferreira Filho, Virgílio José Martins *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Produção. III. Título.

“ A Deus toda glória.”

Agradecimentos

A Deus por guiar os meus caminhos e me oferecer tantas oportunidades de crescer.

Aos meus pais Lidia Maria e José Carlos por me ajudarem desde a minha infância.

Ao meu marido Alex Boucher por escolher estar ao meu lado e me dar suporte aqui no Brasil durante o fim do doutorado.

Aos meus orientadores Virgílio Ferreira Filho e Laura Bahiense pela sua disponibilidade em me aconselhar a qualquer hora, ajudando a conseguir uma grande experiência de intercâmbio no Canadá e me apoiando em situações difíceis no desenvolvimento da tese.

Ao professor Leandro Callegari Coelho por me oferecer orientação para o programa ELAP, me recebendo em Quebec, me acompanhando e dando suporte para a submissão do artigo.

Ao professor Luiz Satoru Ochi, que contribuiu com seus comentários na banca de qualificação, ao professor Edilson Fernandes Arruda, que contribuiu com o primeiro projeto realizado e ambos por aceitarem fazer parte da banca examinadora da defesa de doutorado.

Ao professor Glaydston Ribeiro, que além de participar da banca, disponibilizou um pouco do seu tempo para me aconselhar sobre possíveis melhorias em meu trabalho durante a elaboração.

A minha orientadora da graduação Professora Christina Waga por me incentivar a cursar o Mestrado na COPPE/UFRJ e sempre prosseguir. Ao professor Samuel Jurkiewicz pela sua receptividade e disponibilidade durante todo o curso de Mestrado e Doutorado, especialmente no início.

As pessoas que fizeram parte dessa caminhada incluindo professores, alunos e funcionários que trabalham ou que passaram pelo Programa de Engenharia de Produção na COPPE.

Ao CNPq pelo auxílio financeiro concedido, que foi de fundamental importância para o desenvolvimento deste trabalho.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

UM ALGORITMO DE BUSCA HÍBRIDO PARA O PROBLEMA DE ROTEAMENTO DE EMBARCAÇÕES DE SUPRIMENTO PERIÓDICO

Juliana Beatriz Carvalho de Oliveira Soares Boucher

Maio/2018

Orientadores: Virgílio José Martins Ferreira Filho
Laura Silvia Bahiense da Silva Leite

Programa: Engenharia de Produção

O transporte marítimo de petróleo de alta qualidade é fundamental para assegurar um fluxo de mercadorias dentro do prazo previsto, reduzindo o custo total da logística e garantindo operações eficientes. Nesta tese, o focus está no transporte de carga de convés para unidades marítimas como observado nas operações do parceiro industrial no Rio de Janeiro, Brasil. O principal objetivo desta pesquisa é definir as rotas marítimas para resolver um problema de roteamento de embarcações de suprimento periódico, levando em consideração a programação do porto e o horário de abertura em algumas instalações. Os procedimentos atualmente utilizados pela empresa são descritos, e o problema é matematicamente formulado. Dado que o tamanho das instâncias é muito grandes para serem resolvidos exatamente, propõe-se diferentes métodos visando alcançar melhores soluções com um baixo tempo computacional. O primeiro método, composto por três fases, usa uma heurística de agrupamento de unidades marítimas combinada a um método exato a fim de realizar o roteamento das mesmas e finaliza com o sequenciamento das rotas de acordo com as restrições de horários de funcionamento, partidas do porto e tempo entre atendimentos. O segundo método utiliza a heurística *adaptive large neighborhood search* (ALNS) na tentativa de reduzir a quantidade de operações realizadas pela heurística anterior e o tempo computacional. Por fim, propõe-se um método híbrido baseando-se na heurística ALNS e inovando com conceitos do algoritmo *clustering search* que propõe uma detecção de regiões promissoras do espaço de busca. Os resultados computacionais indicam que a heurística híbrida traz benefícios ao ALNS ao encontrar melhores soluções em menos tempo e ainda reduz o coeficiente de variação dentro de uma amostra de soluções em diferentes execuções.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

A HYBRID SEARCH ALGORITHM FOR PERIODIC SUPPLY VESSELS ROUTING PROBLEM

Juliana Beatriz Carvalho de Oliveira Soares Boucher

May/2018

Advisors: Virgílio José Martins Ferreira Filho
Laura Silvia Bahiense da Silva Leite

Department: Production Engineering

High quality maritime petroleum transportation is critical to ensure timely flow of goods, reducing the total logistics cost and guaranteeing an efficient process. In this thesis, we focus on the transportation of deck cargo to offshore units as observed in the operations of our industrial partner in Rio de Janeiro, Brazil. The main objective of this research is to define the maritime routes to solve a periodic supply vessels routing problem, taking into account the port departure and opening hours at the offshore facilities. We describe the solution procedures currently used by the company, and we formally formulate the problem mathematically. Given that the sizes of the instances are too large to be solved exactly, we propose different methods to achieve better solutions with a reduced computational time. The first method, composed of three phases, uses a clustering heuristic combined with an exact method in order to perform the routing and ends with a resenquencing according to the operating hours, port departures and time between services constraits. The second method uses the adaptive large neighborhood search (ALNS) heuristic in an attempt to reduce the number of operations performed by the previous heuristic and the computational time. Finally, a hybrid method is proposed based on the ALNS heuristic and innovates with the concepts of the clustering search algorithm that proposes a detection of promising regions of the search space. The computational results indicate that the hybrid heuristic brings benefits to the ALNS by finding better solutions in less time and still reduces the coefficient of variation within a sample of solutions in different executions.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

AI	Agrupamento iterativo, p. 41
ALNS	Adaptive large neighborhood search, p. 3
ANP	Agência Nacional de Petróleo, Gás Natural e Biocombustíveis, p. 2
BL	Módulo de busca local, p. 41
CS	Clustering Search, p. 3
ECS	Evolutionary Clustering Search, p. 13
E&P	Exploração e produção de óleo e gás, p. 1
GRASP	Greedy Randomized Adaptive Search Procedure, p. 13
LNS	Large neighborhood search, p. 11
MA	Módulo de análise, p. 41
MB	Metaheurística de busca, p. 41
OPEP	Organização dos Países Exportadores de Petróleo, p. 1
PIM	Programação inteira mista, p. 13
PRESP	Problema de Roteamento de Embarcações de Suprimento Periódico, p. 3
PRVC	Problema de roteamento de veículo capacitado, p. 24
PRVP	Problema de roteamento de veículos periódico, p. 5
PRV	Problema de roteamento de veículos, p. 5
PR	Path-relinking, p. 44
PSVs	Platform Supply Vessels, p. 2

UC	unidades operacionais da classe de Campos, p. 16
UM	Unidade marítima, p. 2
UR	unidades operacionais da classe do Rio de Janeiro, p. 16
VLNS	Very large neighborhoods search, p. 29
VNS	Variable Neighbourhood Search, p. 13

Capítulo 1

Introdução

Os investimentos tecnológicos são essenciais para melhorar os processos que compõem a indústria do petróleo, por isso essas empresas buscam métodos para aumentar sua eficiência visando a minimização de custos. As atividades de exploração e produção de óleo e gás (E&P) no setor *offshore*, especificamente na camada pré-sal, têm imposto grandes desafios relacionados não apenas a tecnologias voltadas para engenharia de petróleo, mas também ligados à área de planejamento e operação de recursos e infraestrutura logísticos, fundamentais para a continuidade operacional em E&P.

Como mercadoria de importância estratégica, o petróleo apresenta um caráter de extrema relevância para um determinado país ou região, já que seus derivados como gasolina e diesel são utilizados pela maior parte dos meios de transporte. Poucas mercadorias tornaram-se tão importantes quanto o petróleo, uma vez que pode ser usado como fonte de energia, bem como matéria-prima na fabricação de plásticos e fertilizantes [2].

Segundo " BP Energy Outlook - 2018 edition ", petróleo e gás juntos representam mais da metade da energia mundial e prevê-se que a tendência global para o aumento da urbanização continue, com quase 2 bilhões de pessoas adicionais vivendo em centros urbanos até 2040, uma taxa ligeiramente mais rápida de urbanização do que no passado. Além disso, o setor de transporte continua a ser dominado pelo petróleo, apesar da crescente penetração de combustíveis alternativos, particularmente gás natural e eletricidade.

A demanda contínua e crescente de energia de baixo custo e a disponibilidade de recursos de hidrocarbonetos colocam ainda o petróleo como uma importante fonte não-renovável da matriz energética mundial para as próximas décadas do século XXI. A avaliação feita pela Agência Internacional de Energia (International Energy Agency – IEA 2018) prevê que a produção brasileira de petróleo vai ter o maior crescimento do mundo nas próximas duas décadas entre países fora da Organização dos Países Exportadores de Petróleo (OPEP) .

De acordo ainda com a Agência Nacional de Petróleo, Gás Natural e Biocombustíveis (ANP) em dezembro de 2017, a produção de petróleo e gás no pré-sal brasileiro atingiu, pela primeira vez, mais da metade da produção nacional. A produção total do Brasil foi de 3,325 milhões em barris de óleo equivalente por dia (b/d, soma das produções de óleo e de gás natural), sendo 1,685 milhão de b/d (50,7%) do pré-sal.

Portanto, a logística de apoio *offshore*, também chamada de logística de E&P ou apenas logística *offshore* − cujo papel é transportar cargas e pessoas entre instalações em terra, tais como bases de apoio e aeroportos, e unidades marítimas (UMs) , a exemplo, sondas, plataformas e embarcações especiais − surge como uma área com margem relevante para o desenvolvimento de soluções baseadas, por exemplo, em modelos para planejamento e operação de recursos e infraestrutura de forma mais eficiente. Comumente, o transporte marítimo é realizado por embarcações ou barcos chamados *Platform Supply Vessels* (PSVs) . Já a troca de tripulações e o transporte de pessoal especializado são feitos geralmente via helicópteros.

Especificamente sobre as embarcações supridoras, o dimensionamento e a utilização adequados destas são fundamentais para manutenção de uma operação eficiente e com custo aceitável, dado que a quantidade de cargas movimentada é vultosa, tanto no sentido de entrega às unidades, como na remoção para tratamento em terra e reuso *offshore*. O foco deste trabalho é na programação de PSVs para o transporte de carga de convés para as unidades offshore.

1.1 Motivação e objetivo

Os problemas que lidam com roteamento e programação de embarcações são diferentes dos outros modos de transporte porque eles operam sob diferentes condições e exigem modelos de suporte à decisão que se encaixam nas características do problema específico [3]. Este problema foi escolhido devido à sua relevância no mundo real, ao acesso a instâncias reais e ao desafio de desenvolver um estudo que contribua para os avanços científicos nesta área acadêmica. De acordo com FREI [4] "*o petróleo poderá alimentar mais de 80% do setor de transporte global nos próximos 40 anos devido ao forte crescimento da demanda para o setor heavy-duty, o transporte marítimo e o tráfego aéreo. Em 2050, nossos relatórios de projetos que alimentam a demanda global em todos os modos de transporte poderão aumentar de 30% para 82% em relação aos níveis de 2010*".

Outro motivo importante é que nosso estudo é feito através de instâncias reais relacionadas à Bacia de Campos, que é a principal área sedimentar já explorada na costa brasileira. Ela se estende das imediações da cidade de Vitória (Espírito Santo) até Arraial do Cabo, no litoral norte do Rio de Janeiro, em uma área de aproxi-

madamente 100 mil quilômetros quadrados. O transporte marítimo de carga nesta Bacia é uma das operações mais importantes no Brasil de atividades de exploração e produção de petróleo em alto mar.

Em resumo, o objetivo principal desta tese é desenvolver formas de definir boas rotas para o transporte de carga de convés para as unidades *offshore*. Para tanto, o problema é modelado como um problema de roteamento de embarcações de suprimento periódico (PRESP) com restrições de horário de funcionamento para algumas instalações, tempo máximo entre serviços e partidas do porto flexíveis, porém já definidas. Para alcança-lo de forma eficiente, têm-se como objetivos secundários:

1. descrever as operações envolvendo o transporte de carga de convés para as unidades offshore;

2. modelar o problema por meio de programação linear inteira mista, passível de ser resolvido por método exato;

3. desenvolver uma heurística de agrupamento e roteamento elaborada a fim de encontrar soluções viáveis dividindo o problema em grupos menores;

4. desenvolver uma heurística *adaptive large neighborhood search* (ALNS) trazendo rapidez na busca de soluções;

5. desenvolver uma heurística híbrida que inclui conceitos da heurística ALNS e do método de pesquisa em grupos de soluções (CS, do inglês *clustering search*) objetivando obter boas soluções em baixos tempos computacionais;

6. finalizando os objetivos secundários dessa tese está a comparação dos resultados das diferentes abordagens para resolver o problema.

A heurística híbrida é a contribuição principal dessa tese. Neste método, a busca é intensificada apenas em regiões do espaço de soluções que são consideradas promissoras. A ideia de detectar regiões promissoras é justificada pelo fato de que a busca local é realizada exclusivamente em áreas verdadeiramente promissoras, reduzindo assim o número de chamadas à função objetivo que, para problemas reais, pode ter um custo significativo. Desta forma, obtêm-se uma melhoria na velocidade de convergência, bem como uma diminuição do esforço computacional na aplicação da busca local que será realizada de forma mais racional.

1.2 O problema de programação de barcos para a logística offshore

A logística offshore pode ser definida como o conjunto de atividade necessárias para garantir o suprimento de insumos para as unidades operacionais no mar. Temos como base a logística usada pela companhia de petróleo brasileira parceira que cedeu seus dados para as pesquisas realizadas. Especializada na indústria de óleo, gás natural e energia, essa companhia está presente nos segmentos de exploração e produção, refino, comercialização, transporte, petroquímica, distribuição de derivados, gás natural, energia elétrica, gás-química e biocombustíveis. A maior parte das reservas de petróleo está em campos marítimos, águas profundas e ultraprofundas. Seus espaços de atuação localizam-se nas Bacias de Campos, Santos, Espírito Santo, Solimões, Potiguar, Sergipe e Alagoas, Recôncavo, Camamu-Almada, Tucano e Jequitinhonha. O abastecimento das plataformas é feito com vários tipos de carga, tais como carga geral, equipamentos, alimentos e água potável, tubos, risers, produtos químicos e resíduos alimentares.

A logística de suprimento de carga de convés na empresa do caso de estudo baseia-se na construção de planos de visitas às UMs. Nestes planos, as rotas realizadas pelos PSVs são definidas e formam um grupo de UMs que demandam carga geral, comumente dividida em pedidos de entrega às UMs e pedidos de devolução delas, as cargas de retorno. Essa demanda acaba por impor operações dos PSVs em uma base de apoio, localizada em um porto, para carregar os pedidos de entrega e descarregar os pedidos de devolução. Tais operações das embarcações em bases de apoio acontecem nos chamados "berços", que são os locais em que a embarcação atraca, e a partir dos quais guindastes movimentam a demanda de cargas entre o convés do barco e o porto.

O problema fundamental estudado nesta tese pode ser definido como: dado um conjunto de clientes que geram pedidos de carga de convés para entrega em certo horizonte de planejamento por um conjunto de viagens, deseja-se encontrar quais pedidos serão entregues em cada viagem; a quantidade de produtos carregados em cada veículo em cada viagem; e os tempos que os veículos levarão para operar em cada locação (UM ou porto). Como objetivo, visa-se a minimização dos custos com viagens às UMs. Como principais restrições, têm-se a capacidade de cada barco e os prazos de entrega das cargas. Duas das principais características do problema são os horários de funcionamento das plataformas e o tempo entre as visitas, chamado de tempo de ciclo. Esse período em que a UM deve ser servida é administrado de maneira que o atendimento seja realizado duas vezes por semana, distribuindo sua demanda uniformemente entre as visitas. Assim, dada uma programação do porto com os horários de saída para as embarcações, é necessário construir rotas para

as quais a sequência de suprimento deve considerar as UMs que operam somente durante o dia ou a noite repeitando o intervalo máximo entre visitas. Devido às estratégias administrativas feitas para facilitar a programação das embarcações, a empresa procura manter as mesmas UMs em duas viagens subsequentes de uma embarcação. No entanto, é importante garantir que a embarcação saia do porto seguindo um determinado cronograma e chegue a uma instalação aberta ou que ofereça o menor tempo de espera possível. Além disso, o tempo de ciclo para cada UM não deve ser muito longo ou muito curto devido ao planejamento que visa dois abastecimentos por semana.

1.3 Problemas de roteamento de veículos periódico e métodos de solução

O problema descrito na seção anterior pode ser modelado como problema de roteamento de veículos periódico (PRVP) , especificamente voltado para embarcações, incluindo restrições de horários de funcionamento de determinadas UMs e partidas do porto, sendo chamado de PRESP. O PRVP é uma generalização do problemas de roteamento de veículos (PRV) em que rotas de veículos são construídas por um período de tempo. Para solucionar o PRESP de forma eficiente, foram pesquisados vários métodos de solução na literatura que lidam com PRVP ou algumas extensões do PRV e estratégias para unificá-los. Sendo assim, estudamos heurísticas significativas para este trabalho, incluindo heurísticas clássicas, metaheurísticas e métodos baseados em programação matemática. De acordo com a literatura, a dificuldade deste problema está no alto número de soluções disponíveis, considerando que a distância entre clientes é simétrica, o número total de soluções possíveis é $(n-1)!/2$. Além disso o PRVP é classificado na literatura como NP-hard [5] o que significa que não há algoritmos que o resolvem em tempo polinomial quando lida-se com grandes instâncias.

Sendo assim, tais problemas são comumente abordados através de heurísticas e/ou metaheurísticas. De forma geral, esses métodos utilizam combinação de escolhas aleatórias e conhecimento histórico (dos resultados anteriores adquiridos pelo método) para se guiarem e realizar suas buscas em vizinhanças dentro do espaço de soluções, tentando evitar paradas prematuras em ótimos locais. Buscas heurísticas são implementadas visando boas soluções em um curto tempo computacional e são adaptadas ao problema tentando aproveitar ao máximo as particularidades do mesmo, mas geralmente ficam presas em um ótimo local. Metaheurísticas, por outro lado, são técnicas independentes de problemas. Em geral, não são gulosas, podendo até aceitar uma deterioração temporária da solução, o que lhes permite explorar

mais detalhadamente o espaço da busca e obter uma solução melhor.

Para resolver o problema objeto deste trabalho três abordagens foram seguidas: a primeira consiste em agrupar e rotear. em uma fase inicial formam-se grupos de clientes de acordo com uma heurística de agrupamento, a seguir aplica-se um método exato para rotear e por fim re-sequenciar a fim de cumprir o planejamento considerando as horas de abertura das UMs [6]. Esta heurística é capaz de fornecer boas soluções, entretanto lida com muitas etapas e um tempo computacional elevado.

A segunda abordagem chamada de ALNS foi desenvolvida por PISINGER e ROPKE [7] e é capaz de fornecer soluções de alta qualidade dentro de um curto tempo computacional, todavia não tão satisfatório por ser um método de busca que tem dificuldade em escapar de ótimos locais [8]. Por fim, a implementação dos métodos citados anteriormente nos permite ter uma base e compara-los com uma nova heurística híbrida ainda mais eficiente proposta nesta tese. Esta terceira abordagem consiste em utilizar conceitos da heurística ALNS e CS proposta por OLIVEIRA e LORENA [9], resultando em um método híbrido que combina adequadamente metaheurísticas e heurísticas de busca local.

1.4 Organização do texto

Esta tese está dividida da seguinte forma: o **Capítulo 2** fornece uma revisão bibliográfica do PRVP, alguns dos seus métodos de solução e inclui trabalhos encontrados na literatura que mais se aproximam do PRESP. A descrição detalhada do nosso problema está contida no **Capítulo 3** onde são dadas especificações sobre todos os fatores que compõe o problema como o porto, os clientes, as embarcações e o funcionamento completo do sistema logístico de suprimento.

O **Capítulo 4** apresenta a formulação do modelo de programação matemática estruturado para o PRESP, bem como a descrição das heurísticas de agrupamento-roteamento e ALNS implementadas. Além disso, os conceitos do algoritmo base CS são descritos e a heurística híbrida é introduzida. Resultados computacionais podem ser encontrados no **Capítulo 5** relatando procedimentos realizados, calibração de parâmetros e comparações entre as diferentes abordagens para resolução do PRESP. Finalmente o **Capítulo 6** conta com um resumo da tese e inclui resultados alcançados e sugestões para trabalhos futuros.

Capítulo 2

Revisão Bibliográfica

Logística pode ser entendido como transportar produtos para diferentes locais em diferentes tempos. Sua necessidade é muito antiga, por exemplo NOVAES [10] relata que, durante a fase colonial, os pedidos de comerciantes eram feitos por um caixeiro viajante e eles aguardavam por meses até um pedido chegar às estações comerciais. À medida que mais cidades emergiam, expandiu-se o comércio e outras necessidades, como a diversificação de produtos. Estratégias logísticas e soluções tecnológicas tiveram um desenvolvimento extraordinário durante e depois da Segunda Guerra Mundial.

A logística passou a ser considerada uma ferramenta importante que agrega o valor do lugar, tempo, qualidade e informação à cadeia de produção. Atualmente, a logística moderna evita adicionar ao processo tudo aquilo que não tem valor para o cliente, ou seja, que só gera custos e perda de tempo. A ênfase não é apenas na garantia da qualidade do produto, mas também nos serviços associados que procuram sistematicamente reduzir custos em todos os níveis.

Essas mudanças no sistema logístico foram incorporadas na crescente demanda por serviços de valor agregado e na integração de vários modos de transporte. Consequentemente, os serviços logísticos de alta qualidade e a integração eficaz e eficiente dos sistemas de transporte e logística oferecidos por um operador marítimo (isto é, uma companhia de navegação ou operador de porto / terminal) tornaram-se uma questão importante [11].

O conceito de logística envolve essencialmente planejamento para a realização de diversas tarefas nas empresas. Especificamente no ramo de petróleo e gás offshore, sem a logística correta, todo o sistema não funcionará corretamente. Ou seja, para extrair petróleo em águas profundas é necessário um bom planejamento do transporte marítimo.

A cadeia logística compreende o armazenamento e transporte terrestre das cargas, a operação portuária e o transporte marítimo até as plataformas, além de atividades como transporte de trabalhadores e serviços como movimentação de uni-

dades marítimas. O planejamento e gerenciamento das atividades de transportes de cargas são essenciais, pois além de viabilizar o fluxo de produtos ao longo da cadeia de suprimentos, os custos associados ao transporte representam, em média, de um a dois terços dos custos logísticos totais em uma empresa [12].

Dessa forma, em tais sistemas, qualquer melhoria que seja obtida na área de suprimentos é capaz de provocar fortes impactos na redução dos custos globais. Portanto, é importante tentar minimizar os custos de transporte, tanto quanto possível, por meio do bom planejamento das embarcações de abastecimento de uma forma eficiente.

2.1 Logística offshore

A cadeia de petróleo e gás pode ser dividida em "upstream", "midstream" e "downstream". As atividades que visam servir as instalações offshore com os estoques necessários são denominadas logística "upstream", ao passo que logística "downstream"engloba as atividades que visam levar os derivados do petróleo e gás aos clientes finais. As operações "midstream" ligam as logísticas "upstream"e "downstream" , incluindo principalmente transporte e armazenamento de recursos, como dutos e sistemas de coleta. De acordo com KAISER e SNYDER [13], "*...o estudo da logística da indústria de petróleo e gás offshore "upstream" tem sido diversificado, mas não teoricamente unificado ou bem desenvolvido...*". Esses estudos abordam tópicos como pesquisa operacional aplicada a sistemas de logística offshore, gerenciamento de informações, terceirização e outros.

As abordagens para o roteamento de embarcações de abastecimento para servir as unidades offshore podem ser classificadas em várias categorias: envolvendo recolhimento e entrega, frota homogênea ou heterogênea, bens múltiplos, incertezas, prioridade de algumas cargas, viagens múltiplas, problemas climáticos e restrições como a área do convés da embarcação e os limites de tempo. Além disso, existem problemas que podem ser diretamente relacionados ao roteamento de embarcações de abastecimento quanto ao número de berços no porto [14] e sobre transporte aéreo offshore [15].

Os problemas de roteamento e programação de embarcações são diferentes dos outros modos de transporte porque os barcos operam sob diferentes condições e requerem modelos de suporte à decisão que atendam às características do problema específico [3]. Neste tipo de problema, é necessário adicionar as janelas de tempo resultantes do fato de que alguns clientes impõem prazos de entrega e restrições de tempo de entrega. Além da distância total de viagem, a função objetivo deve incluir uma penalidade para espera quando um barco chega muito cedo para atender um cliente [16].

2.2 Problema de Roteamento de Veiculos Periódico

O problema trabalhado nesta tese aborda o PRVP com restrições operacionais relacionadas a programação do porto e horas de abertura de instalações offshore. O PRVP é um assunto frequentemente estudado em pesquisa operacional e inclui referências notáveis ([17–20]). A fim de criar uma programação cíclica ou periódica, esses artigos descrevem frequências de serviços e modelos garantindo uma boa distribuição de suprimento aos clientes.

O PRVP é uma generalização do PRV, em que as rotas do veículo são construídas por um período de tempo e foi introduzido por BELTRAMI e BODIN [21] em 1974 para alocar caminhões compactadores de lixo na coleta de lixo municipal. Este artigo contribuiu significativamente para o surgimento de diferentes aplicações nos últimos anos. Seu objetivo principal não era formular ou definir o problema formalmente, apesar de proporem heurísticas para resolvê-lo. Eles mostraram a dificuldade do PRVP quando comparado ao padrão PRV. Poucos anos depois, é fornecido por RUSSELL e IGO [22] a definição formal do PRVP como um problema de roteamento por atribuição. Este artigo procura resolver as dificuldades de escolher um cronograma para cada nó e considerar o problema de roteamento de alocação como um problema inteiro misto, impondo restrições à capacidade do veículo, bem como a duração máxima de qualquer rota.

A primeira formulação do PRVP foi apresentada por CHRISTOFIDES e BEASLEY [23] e, com o objetivo de encontrar a frequência exigida de visitas ao cliente, eles o definem como um problema de planejar um conjunto de rotas para cada dia de um determinado período de dias. A formulação de programação inteira usa conjuntos de variáveis de decisão para o roteamento de um veículo em um dia específico e para a atribuição de clientes aos horários. Os métodos de solução nestes e artigos subsequentes se concentraram em heurísticas de dois estágios (construção e melhoria);vide TAN e BEASLEY [24] e RUSSELL e GRIBBIN [25].

2.2.1 Métodos de resolução encontrados na literatura para o PRVP

Os algoritmos exatos para qualquer tipo de PRV envolvem desde métodos básicos até programação matemática de alta complexidade [26]. No entanto, pertencente à categoria conhecida como NP-hard, PRVP tem ordem de complexidade exponencial [27] dificultando a resolução exata do problema com dados reais. FRANCIS *et al.* [28] desenvolvem um método de solução exata baseado no relaxamento lagrangiano de uma formulação de programação inteira do PRVP na qual a frequência de

atendimento é uma decisão do modelo e a escolha da quantidade de serviços não é pré-definida, e sim minimizada pelo modelo. MOURGAYA e VANDERBECK [29] resolvem uma versão tática do PRV multi-período na qual os horários das visitas e as atribuições de clientes aos veículos são tratados simultaneamente através de uma heurística baseada em geração de colunas.

Quando se trata de casos reais, a maioria dos estudos são construídos em base heurística ou metaheurística. Métodos semelhantes aos de BELTRAMI e BODIN [21] que tratam de duas fases, são comumente encontrados em heurísticas para este problema. RUSSELL e IGO [22] apresentam uma heurística de melhoria e duas heurísticas de construção trabalhando com o problema de coleta de resíduos com clientes que exigem serviço diário e em dias fixos. Na primeira heurística, eles usam métricas que relacionam a distância média do nó para fazer a alocação de nós a dias. Na fase seguinte, uma etapa de melhoria tenta reatribuir nós a outros horários após a construção inicial. Para obter melhores resultados, eles desenvolvem uma heurística de melhoria que reotimiza a alocação e roteamento de nós para um segundo lugar. A terceira heurística é uma implementação do método de economia Clarke-Wright, garantindo que qualquer movimento de economia proposto resulte em uma alocação viável de nós para os dias.

TAN e BEASLEY [24] determinam a alocação de nós para determinadas programações de atendimento usando um problema de atribuição, feito isso, resolvem PRVs independentes para cada um dos dias. Os autores propõem encontrar a medida de custo através da avaliação de rotas, que seria uma boa representação do custo real de servir as rotas nos dias associados ao cronograma. GAUDIOSO e PALETTA [30] impõem restrições sobre a duração máxima da rota, a capacidade do veículo e restrições sobre o número mínimo e máximo de dias entre visitas para cada nó. Eles sugerem uma heurística alternativa para o problema de minimizar o tamanho da frota, diferente do problema operacional usual de reduzir a distância. O objetivo é agendar as entregas de acordo com combinações viáveis de dias de entrega e determinar as políticas de agendamento e roteamento dos veículos, a fim de minimizar ao longo do horizonte de planejamento o número máximo de veículos empregados simultaneamente, ou seja, o tamanho da frota.

Métodos heurísticos ou metaheurísticos construídos para resolver um determinado problema podem escapar da armadilha da otimização local que afeta as heurísticas convencionais. Isso decorre do fato de que, sendo abordagens intuitivas, a estrutura particular do problema pode ser analisada de forma inteligente, obtendo uma solução adequada para o problema. CHAO *et al.* [27] desenvolvem um método metaheurístico que cria uma solução viável inicial usando a formulação de CHRISTOFIDES e BEASLEY [23] e, em seguida, usa etapas de melhoria para avançar na solução ideal para o PRVP. Ao minimizar a carga máxima carregada em qualquer

dia, eles tentam resolver um relaxação linear do problema de atribuição de alocação de nós aos dias de entrega.

CORDEAU *et al.* [17] apresentam uma heurística de busca tabu para resolver três problemas de roteamento bem conhecidos, incluindo o PRVP. A contribuição deste trabalho foi o desenvolvimento de uma busca tabu simples que contém muito poucos parâmetros controlados pelo usuário. NGUYEN *et al.* [31] propuseram um algoritmo genético híbrido baseado nos conceitos de algoritmo genético e heurísticas de busca local. Seguindo esta ideia, DRUMMOND *et al.* [32] desenvolvem uma heurística de mecanismo de população de thread paralelo que é uma implementação de algoritmos genéticos em uma estrutura de computação paralela, juntamente com métodos de busca locais modificados. VIDAL *et al.* [19] propõem uma metaheurística baseada no algoritmo genético para resolver o problema de roteamento de veículos multi-depósito e periódico, incluindo uma série de recursos avançados em termos de avaliação das soluções, geração de descendentes e gerenciamento da população, contribuindo para uma boa performance.

2.2.2 Heurísticas aplicadas ao PRESP

O PRESP é significativamente mais complexo que o clássico PRVP porque as embarcações lidam com múltiplas viagens e conta ainda com a presença de outras características como as horários de funcionamento diurnas ou noturnas e as restrições de saídas do porto flexíveis. Portanto a literatura para tal problema é bastante escassa.

FAGERHOLT e LINDSTAD [33] apresentam uma versão simplificada do PRESP sem atribuição de viagens a dias e tratam as restrições de partidas com a frota de embarcações necessárias e suas respectivas viagens. HALVORSEN-WEARE *et al.* [34] apresentam o método de solução baseado nas viagens (*voyage-based*) e que consiste em duas fases. A primeira fase gera todas as viagens candidatas que os navios podem navegar e a segunda resolve o modelo baseado na viagem. HALVORSEN-WEARE e FAGERHOLT [35] introduziram recentemente um modelo de fluxo de arcos alternativo ao *voyage-based*, mas que não obteve a mesma performance.

A heurística *large neighborhood search* (LNS) é apresentada por SHYSHOU *et al.* [36] para resolver o problema de planejamento de embarcações de abastecimento periódico e o problema de composição da frota. Eles demonstram que a heurística é bastante eficiente para as instâncias consideradas e pode ser usada para resolver o que chamam de grandes instâncias (31 instalações) dentro de um prazo razoável. Geralmente, o método LNS constrói movimentos de vizinhança que fazem pequenas mudanças na solução atual. Esses tipos de métodos são capazes de investigar inúmeras soluções em um curto espaço de tempo, mas provavelmente quando apli-

cados a problemas muito restritos ou problema com base em dados reais podem ter dificuldades em mudar de uma área promissora do espaço de solução para outra [37].

BORTHEN *et al.* [38] adaptaram a heurística de busca baseada no algoritmo genético de VIDAL *et al.* [19] por meio de uma simplificação do PRESP considerando partidas do porto únicas e fixas e sem janelas de tempo.

2.3 Potencias heurísticas para a resolução do PRESP

Uma heurística que pode ser aplicada a uma grande classe de problemas de otimização difíceis é a heurística ALNS descrita por PISINGER e ROPKE [7]. A estrutura do ALNS baseia-se no algoritmo LNS apresentado por SHAW [39], no qual algumas sub-heurísticas concorrentes são escolhidas para modificar a solução atual. Um algoritmo é escolhido para destruir a solução atual em cada iteração e depois outro é escolhido para reparar a solução. Finalmente, se a nova solução satisfizer alguns critérios definidos pela estrutura de busca local, ela é aceita.

Esta heurística pode basear-se em critérios diferentes para a busca local, por exemplo, recozimento simulado, busca tabu, busca local guiada e critérios de aceitação simples, como aceitar todas as soluções de melhoria. Vários tipos de problemas relacionados ao problema de roteamento foram resolvidos através de ALNS e heurísticas similares, obtendo resultados positivos. Recentemente, KISIALIOU *et al.* [40] apresentam o PRESP com embarcações acopladas e aplicam a heurística ALNS sugerindo resultados ainda melhores para as instâncias tratadas por SHYSHOU *et al.* [36].

O ALNS já foi adaptado a problemas de transporte, incluindo roteamento de veículos [37], roteamento de arcos [41], problema de roteamento de plataforma de trabalho [42] e roteamento com estoques [43]. ROPKE e PISINGER [37] apresentam esta heurística para resolver o problema de roteamento de veículos com janelas de tempo. Eles testam em mais de 350 instâncias de referência com até 500 pedidos de clientes e os testes indicam que é vantajoso usar várias sub-heurísticas concorrentes em vez de apenas uma. O problema de roteamento de veículos capacitado acumulado foi estudado por RIBEIRO e LAPORTE [44] e a heurística foi aplicada a um conjunto de instâncias de referência e comparada a dois algoritmos meméticos publicados.

GHILAS *et al.* [45] usam ALNS devido à complexidade do problema de coleta e entrega com janelas de tempo e linhas agendadas em que o conjunto de veículos serve pedidos de frete usando o transporte disponível. Na maioria dos casos, o algoritmo ALNS gerou as mesmas soluções que o CPLEX, mas em um tempo de CPU significativamente menor. MULLER e SPOORENDONK [46] apresentam uma

heurística ALNS híbrida que usa programação inteira mista (PIM) na fase de reparo da heurística ALNS para o problema multi-item de dimensionamento dinâmico capacitado. A PIM repara vizinhanças usando a solução atual como um limite superior inicial e nunca retorna uma solução pior do que a atual. O problema de roteamento do veículo com frota heterogênea e janelas de tempo é resolvido por KOÇ *et al.* [47] através de uma metaheurística chamada algoritmo híbrido evolutivo. Eles usam ALNS equipados com uma gama de operadores como o principal processo de aprendizado dentro da pesquisa.

Outra heurística promissora é o CS. Este método foi proposto como uma forma genérica de combinar metaheurísticas de busca com agrupamento visando detectar áreas promissoras antes de aplicar procedimentos de buscas locais [48]. A pesquisa é intensificada apenas em regiões do espaço de busca que são consideradas promissoras. As ideias originais por trás do Evolutionary Clustering Search (ECS) , propostas em [9], foram validadas por uma aplicação bem sucedida em problemas de otimização numérica sem restrições. A principal diferença é que no processo de agrupamento o ECS executa-se simultaneamente em um algoritmo evolutivo, identificando grupos de indivíduos que merecem atenção especial, enquanto que no CS o algoritmo evolutivo é substituído por metaheurísticas distintas, como recozimento simulado (SA, do inglês *Simulate anneling*), *Greedy Randomized Adaptive Search Procedure* (GRASP) , busca tabu, entre outros.

Posteriormente, em problemas de roteamento, a busca guiada por agrupamento foi estendida a um GRASP com Variable Neighbourhood Search (VNS) por CORRÊA e CHAVES [49] sendo aplicado para o problema do caixeiro viajante *prize-collecting*. Nesse problema, um vendedor coleciona um prêmio em cada cidade visitada e paga uma penalidade por cada cidade não visitada, considerando os custos de viagem entre as cidades. O objetivo é minimizar a soma dos custos de viagem e as penalidades pagas, incluindo no turno cidades suficientes para colecionar um prêmio mínimo, definido a priori.

OLIVEIRA *et al.* [51] consideram o CS com SA para o problema de alocação do ancoradouro. RIBEIRO *et al.* [52] aplicam o CS para o problema de roteamento de sondas de manutenção, que é uma variante do problema de roteamento de veículos com janelas de tempo. Os resultados computacionais mostram que o CS é uma boa heurística para grandes instâncias.

Capítulo 3

Descrição do problema

As operações de exploração e produção marítima são apoiadas por um sistema logístico e de serviços, chamado suporte marítimo. Este sistema é composto por portos, aeroportos, depósitos, helicópteros e utiliza várias embarcações especializadas para transportar a carga para UMs. O sistema logístico que apoia as operações de exploração e produção engloba principalmente o transporte de carga para unidades marítimas e carga de retorno, armazenamento de materiais e equipamentos no porto e transporte de pessoas. Os custos da operação de frete e o valor da produção de petróleo são muito altos, especialmente o aluguel das UMs destinadas a perfuração de poços que possuem uma taxa média diária de US$ $500,000$ [53]. Assim sendo, é necessário evitar a interrupção dessas atividades devido a atrasos no transporte de carga e destacar a importância do planejamento para minimizar os custos dessas operações.

No caso da companhia parceira, as bacias offshore localizadas no sudeste do Brasil são Santos, Espírito Santo e Campos, como mostrado na Figura 3.1. A bacia de Campos representa cerca de 80% da produção brasileira de petróleo e gás sendo servida principalmente pelo porto de Macaé, o principal porto de apoio marítimo do Brasil que lida com mais de 50% do fluxo de carga de convés e está localizado a 180 km ao norte do Rio de Janeiro [1].

3.1 Depósito e consolidação de carga

A maioria dos depósitos estão localizados nas proximidades de Macaé. A sua responsabilidade é acompanhar o inventário e os pedidos de carga das instalações offshore. Todos os materiais e itens devem passar pelo depósito para serem inspecionados antes de serem enviados para outro depósito ou para o porto. A consolidação da carga é responsável por decidir o modo de transporte, tamanho, destino e prioridade da carga nos depósitos que logo após responde aos pedidos e prepara o trânsito de itens para o porto ou para outros locais.

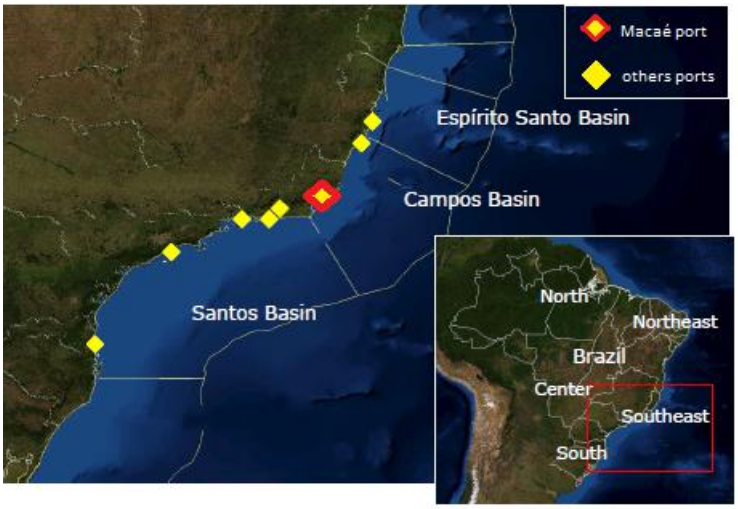


Figura 3.1: Mapa do sudeste do Brasil, incluindo as configurações do porto para suporte as atividades logísticas (Adaptado de LEITE [1])

3.2 Operações portuárias e clientes

As operações portuárias são responsáveis por todas as tarefas após a chegada da carga nos depósitos e atividades de carregamento/descarregamento na proximidade do porto, como mostrado na Figura 3.2. Os tempos de ancoragem de embarcações e alocação de carga também são descritos e usados de acordo com cada necessidade. O tempo de operação no porto e UMs é calculado através de unidades de elevação, ou seja, o número de movimentos do guindaste necessários para transportar a carga para dentro e para fora dos navios. A programação do porto é fornecida e é preciso ajustar os tempos de carregamento específicos para cada rota com os tempos de carregamento no berço, de forma a garantir a eficiência prática do roteamento.

Figura 3.2: Exemplo de atracação no porto de Macaé

Os principais clientes são unidades de produção, plataformas de perfuração e embarcações especiais. As unidades de produção são classificadas em três grupos.

Figura 3.3: Plataforma fixa

Os FPSOs (do termo em inglês *floating, processing, storage and offloading*) que são navios capazes de processar e armazenar o petróleo e providenciar a transferência de petróleo e/ou gás natural. As plataformas semi-submersíveis (SS) que podem ser utilizadas como unidades de produção ou como plataformas de perfuração. Finalmente, as plataformas fixas que são projetadas para receber todos os equipamentos de perfuração, armazenamento de material, alojamento de pessoal, bem como todas as facilidades necessárias para a produção de poços como, por exemplo, na Figura 3.3. As plataformas de perfuração são de tipo: SS ancorado, SS posicionamento dinâmico, *jack-up* e navios de perfuração.

Para evitar grandes mudanças nas operações de rotina, sondas e unidades de produção são roteadas em grupos separados. As UMs são separadas em unidades operacionais das classes de Campos (UC) e Rio de Janeiro (UR) , enquanto que sondas são roteadas separadamente. Ainda existem unidades de manutenção que são agrupadas com as UMs ou sondas, dependendo de onde ocorre a manutenção. Os dados foram coletados da empresa petrolífera parceira e incluem demandas, tempo de serviço, horário de abertura, rotas atualmente realizadas pela empresa, localização das UMs e programação do porto.

3.3 Platform Supply Vessel

As embarcações utilizadas são do tipo PSV, especializado em suporte de sondas ou plataformas de produção. Sua principal função é transportar suprimentos para as plataformas e, geralmente, retornar com cargas para a costa, mas alguns PSVs são usados como petroleiros para transferir óleo diesel para plataformas. Os PSVs podem transportar carga no convés ou em compartimentos de armazenamento abaixo do convés geralmente classificados em toneladas de peso morto (1500, 3000 e 4500).

Para o transporte de produtos básicos que apresentam horários padronizados é utilizado principalmente o PSV 4500, enquanto o PSV 1500 e o PSV 3000 são utilizados para atender a demanda mais flutuante. As custos diários são aproximadamente $US\$25.000$ e $US\$35.000$ para PSV 1500 e PSV 3000, respectivamente [54].

3.4 Características gerais

Todas as instalações têm um tempo de serviço determinado para operações de carregamento e descarregamento da embarcação. Algumas das instalações offshore podem ter horários de abertura e são fechadas para descarregamento e carregamento durante o dia (das 7h às 19h) ou a noite (19h às 7h). Devido às estratégias administrativas para facilitar a programação das embarcações, a empresa procura manter as mesmas UMs para as rotas criadas nas duas viagens subsequentes. No entanto, é importante garantir que a embarcação deixe o porto em um determinado horário e chegue a uma instalação aberta ou com menor tempo de espera possível.

Além disso, o tempo entre visitas semanais em cada UM não deve ser muito longo ou curto. Por exemplo, considere a Figura 3.4. Suponha que uma instalação requer duas visitas durante uma semana e as embarcações de suprimento que visitam a instalação estão agendados para segunda e quarta-feira. Isso significa que haverá um intervalo de dois dias após a primeira viagem, mas serão cinco dias de espera até o serviço recomeçar na próxima semana, o que pode ser muito longo. Por isso, o tempo do ciclo tem limites inferiores e superiores, girando em torno de cerca de 3,5 dias para ambas as visitas nas mesmas semanas e em diferentes semanas. Essa restrição pode ter sua importância ilustrada, por exemplo, pelo fato de uma UM não poder ficar um período maior que o programado para receber carga com alimentos e água potável.

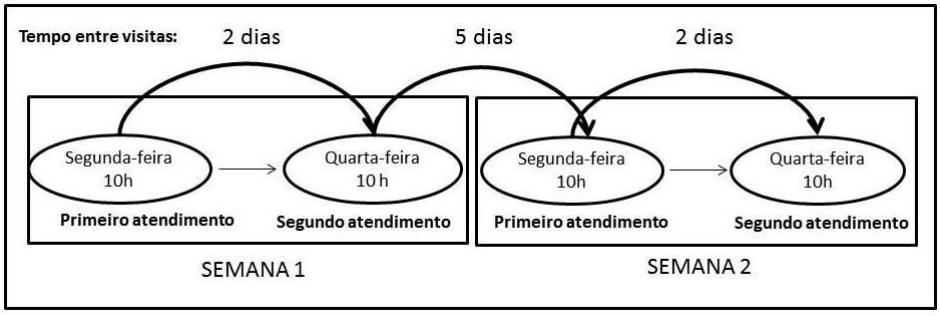


Figura 3.4: Exemplo de como o tempo de ciclo pode ser muito longo com dois atendimentos programados por semana

As demandas das instalações são dadas em m^2 de capacidade do deck e a capacidade total é limitada à área máxima do convés disponível da frota. Há também cargas de retorno que precisam ser transportadas das instalações offshore para o

depósito de abastecimento terrestre, e é por isso que é preciso partir do porto com espaço livre no convés reservado para a primeira carga de retorno. A demanda calculada para cada instalação é a demanda semanal dividida pelo número de visitas. A atribuição de embarcações a rotas é realizada de modo a que o barco possa transportar a carga exigida por cada rota, levando em consideração a demanda média mais a primeira carga de volta, mais um espaço de segurança calculado para garantir um nível de serviço de 90%, assegurando a confiabilidade de todo o processo.

Em resumo, o problema consiste em determinar os horários para as rotas selecionadas e sua respectiva configuração. Uma rota nesta configuração é um grupo de clientes que devem ser atendidos em duas ou três viagens em um planejamento horizontal, tipicamente por semana, com cada viagem começando e terminando no porto. Não existe um cronograma para uma embarcação específica, a companhia de petróleo parceira oferece embarcações de acordo com a necessidade.

Capítulo 4

Metodologias de resolução

O objetivo dos métodos de otimização é acrescentar um pouco de "inteligência"ao processo de enumeração de soluções, reduzindo assim a análise ao espaço de busca e aumentando a possibilidade de resolução de problemas de dimensões mais elevadas.

Esses métodos podem ser classificados como exatos e heurísticos. A garantia de encontrar uma solução ótima para um problema de otimização combinatória é dada pelos métodos exatos [55]. No entanto, para problemas de otimização combinatória que são classificados como NP-hard não existe um algoritmo que consiga provar que uma solução é ótima em tempo polinomial [56]. Por isso, métodos exatos geralmente necessitam de um tempo computacional exponencial, conduzindo assim frequentemente a tempos computacionais inviáveis para o propósito prático.

Em contrapartida, nos métodos heurísticos não se tem a garantia de encontrar soluções ótimas, mas é possível obter boas soluções em um tempo computacional viável [57]. De acordo com CHRISTIANSEN *et al.* [58], a busca da solução de problemas mais realistas é facilitada pelo desenvolvimento rápido de algoritmos de otimização e pelo uso de métodos heurísticos e/ou metaheurísticos substituindo ou associados ao algoritmo exato capazes de alcançar a solução mais próxima ideal em menos tempo e esforço computacional.

Várias dificuldades também podem ser encontradas durante o processo de busca das metaheurísticas, por exemplo quando não é possível determinar se a melhor solução conhecida é um ótimo local ou global, e consequentemente, se a convergência é aceitável. Isto se torna problemático se é levado em conta que a maioria dos problemas de otimização são multimodais, e a ocorrência de vários ótimos locais dificulta o processo de busca.

Evitar a convergência prematura e ao mesmo tempo não impedir que o algoritmo convirja é o grande desafio. Na literatura isto é tratado como diversificação *versus* intensificação [59]. Diversificação significa encontrar novas regiões do espaço de busca que ainda não tenham sido investigadas, enquanto que intensificação significa tentar melhorar a solução corrente realizando pequenas mudanças que conduzam a

novas soluções próximas à solução corrente em uma determinada região. Ao focar em uma dessas duas formas de busca excessivamente pode-se correr o risco de ficar preso em um ótimo local tendo uma velocidade de convergência muito rápida ou levar um alto tempo computacional para encontrar a melhor solução. Sendo assim, é importante balancear a forma de busca tentando identificar rapidamente regiões com soluções de alta qualidade em um tempo computacional reduzido.

Visto isso, ao lidar com problemas reais, a combinação de uma metaheurística com outras técnicas de otimização, chamada metaheurística híbrida, pode proporcionar um melhor desempenho e exploração de sistemas que unam as vantagens das diferentes estratégias.

Neste capítulo apresenta-se as quatro metodologias de resolução aplicadas ao PRESP : a primeira visa construir um modelo exato para o problema. A metodologia seguinte é uma heurística de agrupamento combinada a duas fases que utilizam método exato. Na tentativa de obter um menor tempo computacional e reduzir o número de etapas propõe-se a utilização da heurística ALNS, que lida com operadores de remoção e inserção. Finalmente a heurística híbrida ALNS-CS foi implementada e é descrita como um método de detectar espaços de soluções promissores ao utilizar ideias da heurística CS, visto que a heurística ALNS não possui uma boa variabilidade quando aplicado ao PRESP, especialmente para instâncias de grande porte.

4.1 Programação matemática

Esta secção apresenta um modelo matemático para o PRESP, que é definido por um grafo $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ sendo $\mathcal{V}$ o conjunto de nós e $\mathcal{A}$ o conjunto de arcos. O conjunto de nós $\mathcal{V}$ é definido como $\mathcal{V} = \{0\} \cup \mathcal{C} \cup \mathcal{S}$, o nó 0 representa o porto e o conjunto $\mathcal{C} = \mathcal{P} \cup \mathcal{B}$ representa as unidades marítimas, que são duplicadas artificialmente e particionadas em dois conjuntos: $\mathcal{P} = \{$unidades visitadas na primeira viagem$\}$ e $\mathcal{B} = \{$as mesmas unidades atendidas na segunda viagem$\}$. Define-se também $\mathcal{T} \subseteq \mathcal{C}$ como o conjunto de unidades com restrições de horário de abertura, de tal forma que $\mathcal{T} = \mathcal{D} \cup \mathcal{N}$, sendo $\mathcal{D} = \{$o conjunto de unidades a serem servidas durante o dia$\}$ e $\mathcal{N} = \{$o conjunto de unidades a serem servidas no período noturno$\}$. Finalmente, para permitir diferentes horários de partida do porto 0, criamos um conjunto $\mathcal{S}$ de nós auxiliares, de modo que $\mathcal{S} = \mathcal{S}^1 \cup \mathcal{S}^2$, em que $\mathcal{S}^1$ contém os nós correspondentes à primeira viagem e $\mathcal{S}^2$ contém os nós correspondentes à segunda viagem.

O conjunto de arcos $\mathcal{A}$ contém todos os pares de nós (i, j) tais que $i \neq j,\ i, j \in \mathcal{V}$, e para cada arco (i, j) são conhecidos a distância de viagem d_{ij} e o tempo de viagem t_{ij} que são não negativos. O problema é definido em um horizonte de tempo finito, tipicamente por semana, no conjunto $\mathcal{J}$ de dias. A Figura 4.1 ilustra os conjuntos

que compõe o PRESP.

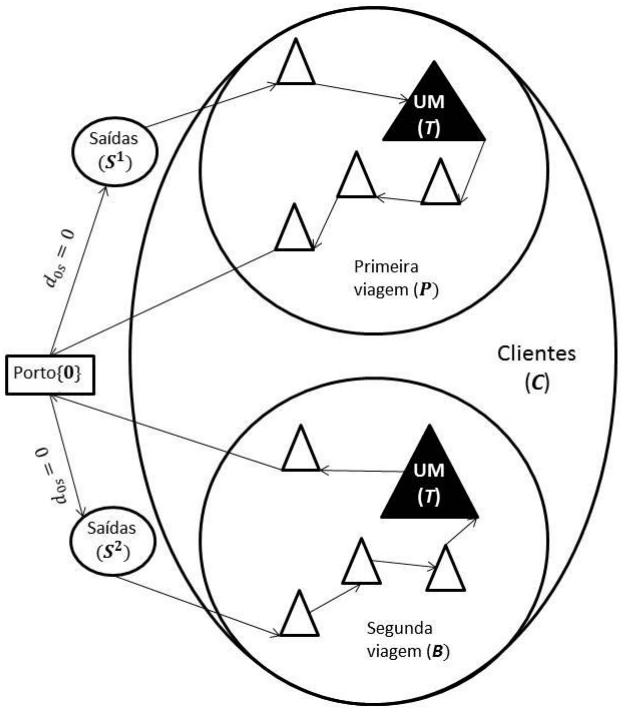


Figura 4.1: Representação dos conjuntos

Cada unidade $i \in \mathcal{C}$ tem associado: um tempo de serviço s_i, quantidade de elevações da carga (lingadas) z_i, quantidade da demanda k_i e carga de retorno ρ_i. Além disso, para cada unidade $i \in \mathcal{T}$ está relacionado um intervalo de horas de abertura $[a_i^d, b_i^d]$ para $d \in \mathcal{J}$. Uma embarcação precisa deixar o porto 0 na hora de partida programada λ_0^d para $d \in \mathcal{J}$ e é permitido chegar à unidade antes que a janela de tempo seja aberta, mas precisa esperar até o início da janela de tempo para o serviço começar. Durante uma semana, cada unidade deve ser visitada duas vezes, então consideramos que um tempo de ciclo entre a primeira e a segunda visita precisa estar no intervalo $[n_{min}, n_{max}]$. R é um tempo máximo de viagem, H representa o tempo máximo de atracação, α e β são coeficientes da regressão usada pela companhia para determinar o tempo de atracação e M é um número suficientemente grande. A capacidade máxima das embarcações é denotada por δ. Finalmente, μ é um peso utilizado na função objetivo para equilibrar seu segundo termo de acordo com a magnitude da soma das distâncias.

Sete tipos de variáveis de decisão são usadas no modelo matemático: $x_{ij}, i, j \in \mathcal{V}$ é uma variável binária que assume o valor um se o arco entre os nós i e j for usado e zero caso contrário; $y_i^d,\ i \in \mathcal{T},\ d \in \mathcal{J}$ é uma variável binária que assume o valor um caso o cliente i seja servido no período $d \in \mathcal{J}$ e zero caso contrário; f_{ij} e g_{ij} são variáveis reais não-negativas que representam respectivamente os fluxos de demanda e lingadas passando pelo arco (i, j) para $i \in \mathcal{V}$ e $j \in \mathcal{C} \cup \mathcal{S}$ para $i \neq j$; os tempos

de espera são representados por w_i para $i \in \mathcal{T}$ e são adicionados à função objetivo sempre que o intervalo é violado; e, finalmente, h_i^1 e h_i^2 são variáveis reais não-negativas que indicam o tempo de chegada no nó $i \in \mathcal{C} \cup \mathcal{S}$ começando a calcular com o barco partindo no horário programado e quando os horários de partida assumem valor zero a fim de calcular o tempo total de viagem, respectivamente.

A Figura 4.2 ilustra os intervalos de tempo usados durante a percurso das embarcações incluindo as diferentes variáveis usadas na formulação do PRESP. O modelo matemático é o seguinte:

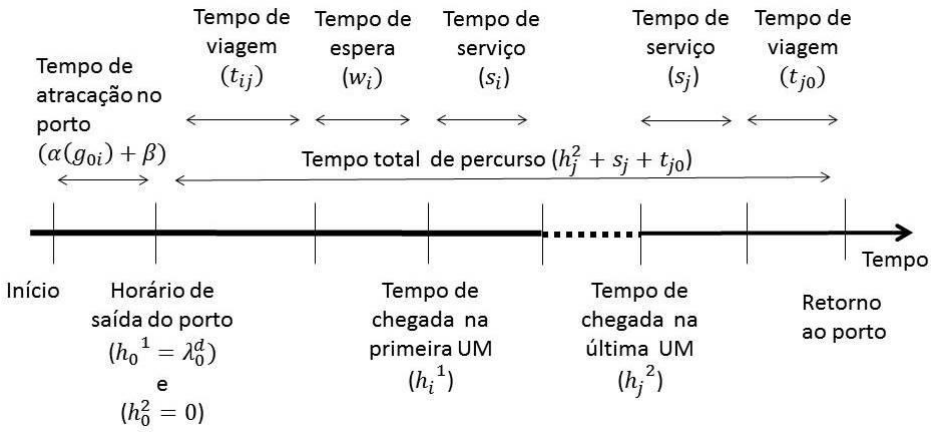


Figura 4.2: Representação das variáveis de tempo

$$\min \sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{V}} d_{ij} x_{ij} \; + \; \mu \sum_{i \in \mathcal{T}} w_i \tag{4.1}$$

s.t.

$$\sum_{j \in \mathcal{C}} x_{ij} = 1 \qquad i \in \mathcal{C} \cup \mathcal{S} \quad (4.2)$$

$$\sum_{i \in \mathcal{C}} x_{ij} = 1 \qquad j \in \mathcal{C} \cup \mathcal{S} \quad (4.3)$$

$$x_{ij} = 0 \qquad i \in \mathcal{C}, j \in \mathcal{S} \quad (4.4)$$

$$x_{ij} = 0 \qquad i \in \mathcal{P}, j \in \mathcal{B} \quad (4.5)$$

$$\sum_{j \in \mathcal{S}} x_{0j} = \sum_{i \in \mathcal{C}} x_{i0} \quad (4.6)$$

$$f_{ij} \leq \delta x_{ij} \qquad i, j \in \mathcal{V} \quad (4.7)$$

$$g_{ij} \leq M x_{ij} \qquad i, j \in \mathcal{V} \quad (4.8)$$

$$f_{ij} \leq (\delta - \rho_j) x_{ij} \qquad i \in \mathcal{S}, j \in \mathcal{C} \quad (4.9)$$

$$\sum_{i \in V} f_{ij} - \sum_{i \in V} f_{ji} = k_j \qquad j \in \mathcal{C} \quad (4.10)$$

$$\sum_{i \in V} g_{ij} - \sum_{i \in V} g_{ji} = z_j \qquad j \in \mathcal{C} \quad (4.11)$$

$$f_{0i} = f_{0j} \qquad i = j, i \in \mathcal{S}^1, j \in \mathcal{S}^2 \quad (4.12)$$

$$g_{0i} = g_{0j} \qquad i = j, i \in \mathcal{S}^1, j \in \mathcal{S}^2 \quad (4.13)$$

$$h_i^1 + w_i + s_i + t_{ij} - M(1 - x_{ij}) \leq h_j^1 \qquad i, j \in \mathcal{C} \quad (4.14)$$

$$h_i^2 + w_i + s_i + t_{ij} - M(1 - x_{ij}) \leq h_j^2 \qquad i, j \in \mathcal{C} \quad (4.15)$$

$$h_j^2 + s_j + t_{j0} - M(1 - x_{j0}) \leq R \qquad j \in \mathcal{C} \quad (4.16)$$

$$\alpha(g_{0i}) + \beta \leq H \qquad i \in \mathcal{S} \quad (4.17)$$

$$\sum_{d \in J} y_i^d = 1 \qquad i \in \mathcal{D} \quad (4.18)$$

$$\sum_{d \in J} y_i^d = 1 \qquad i \in \mathcal{N} \quad (4.19)$$

$$h_i^1 = \lambda_0^d \qquad i \in \mathcal{S}, d \in \mathcal{J} \quad (4.20)$$

$$h_i^2 = 0 \qquad i \in \mathcal{S} \quad (4.21)$$

$$n_{min} \leq h_i^1 - h_j^1 \leq n_{max} \qquad i = j, i \in \mathcal{P}, j \in \mathcal{B} \quad (4.22)$$

$$a_i^d y_i^d \leq h_i^1 \qquad i \in \mathcal{T}, d \in \mathcal{J} \quad (4.23)$$

$$h_i^1 + s_i \leq b_i^d y_i^d + M(1 - y_i^d) \qquad i \in \mathcal{T}, d \in \mathcal{J} \quad (4.24)$$

$$w_i = 0 \qquad i \notin \mathcal{T} \quad (4.25)$$

$$x_{ij}, y_i^d \in \{0, 1\} \qquad i, j \in \mathcal{V}, d \in \mathcal{J} \quad (4.26)$$

$$h_i^1, h_i^2, f_{ij}, g_{ij}, w_i \geq 0 \qquad i, j \in \mathcal{V} \quad (4.27)$$

A função objetivo (4.1) minimiza a soma das distâncias percorridas e a soma ponderada dos tempos de espera nos turnos dia ou noite.

As restrições de atribuição (4.2) e (4.3) garantem que existe apenas um arco entrando e um arco saindo de cada cliente. As restrições (4.4) garantem que os nós auxiliares, que representam o horário de partida, sejam atendidos primeiro. Restrições (4.5) asseguram que existam apenas arcos entre clientes da mesma viagem. A restrição (4.6) estabelece que o número de arcos que saem e entram no porto deve ser o mesmo.

As restrições (4.7) e (4.8) indicam os limites superiores para as demandas e os fluxos de lingadas, respectivamente. Restrições (4.9) garantem espaço livre nos navios para suas primeiras cargas de retorno. As restrições (4.10) e (4.11) definem o fluxo de demanda e lingadas para cada cliente, respectivamente. Restrições (4.12) e (4.13) ligam pares de horários de partida do porto para primeira e segunda viagens através do fluxo de demandas e lingadas representando os mesmos clientes em cada viagem. Restrições (4.14) e (4.15) definem a relação entre os horários de chegada de um cliente e seu sucessor quando os nós auxiliares recebem os horários de partida do porto e quando a partida é feita na hora zero, respectivamente. Essas restrições incluem os tempos de espera quando são necessários, visando respeitar os horários de abertura.

As restrições (4.16) garantem que as rotas tenham um tempo de viagem máximo mais uma folga desde a hora de partida até o retorno do navio para ao porto. Restrições (4.17) limitam o tempo de atracação para evitar atrasos no cronograma

do porto.

A escolha de um dia da semana para atender clientes nos turnos dia e noite é assegurada por restrições (4.18) e (4.19), respectivamente. As restrições (4.20) e (4.21) asseguram que a hora de chegada aos nós auxiliares deve ser igual à janela de início de tempo fornecida pelo cronograma do porto e deve ser igual a zero para marcação de tempo de viagem, respectivamente. As restrições (4.21) garantem que o tempo de ciclo entre a primeira e a segunda viagem deve estar dentro de um determinado intervalo. As restrições (4.22) e (4.23) garantem que o serviço deve ser realizado dentro do período de operação diurna ou noturna. As restrições (4.14), (4.15), (4.16) e (4.24) foram linearizadas introduzindo um número suficientemente grande M. As restrições (4.23) definem tempos de espera nulos para esses nós que não pertencem ao conjunto de unidades com restrições de horário de abertura. Finalmente, as restrições (4.26) e (4.27) aplicam as condições de integralidade e não-negatividade nas variáveis reais binárias e não-negativas, respectivamente.

4.2 Agrupamento-roteamento

Essa heurística mimetiza à metodologia desenvolvida pela companhia de petróleo parceira com uma construção de rotas integrada à programação do porto. Isso acontece porque as operações no porto e o roteamento de embarcações estão conectados isto é, para definir a alocação das embarcações (e suas rotas) nos berços, são necessários os tempos de carregamento, que só são conhecidos após a definição da rota. Por outro lado, a programação do porto é necessária para a construção de rotas que têm restrições de tempo. Nestes casos, para os clientes com restrição de horário de abertura é necessário calcular os tempos de espera e posicionar os clientes nas rotas de maneira que respeitem a limitação de tempo de ciclo.

Para isso, desenvolvemos uma metodologia com 3 fases para resolver o PRESP (vide Figura 4.3): aplicação de uma heurística de agrupamento para dividir a instância em grupos menores (Fase I), resolver o problema de roteamento de veículo capacitado (PRVC) com restrições de tempo máximo de viagem em cada um dos pequenos grupos (Fase II) e, dada a programação do porto, reconstruir a sequência das rotas que contém clientes com restrições de horas de abertura, tempo entre serviços e partidas do porto fixas (Fase III).

4.2.1 Fase 1: Heurística de agrupamento

Na fase I, os clientes são divididos em grupos menores através de uma heurística de agrupamento, com o objetivo de resolver o problema de forma viável e mais rapidamente. O processo começa com uma solução inicial gerada aleatoriamente ou

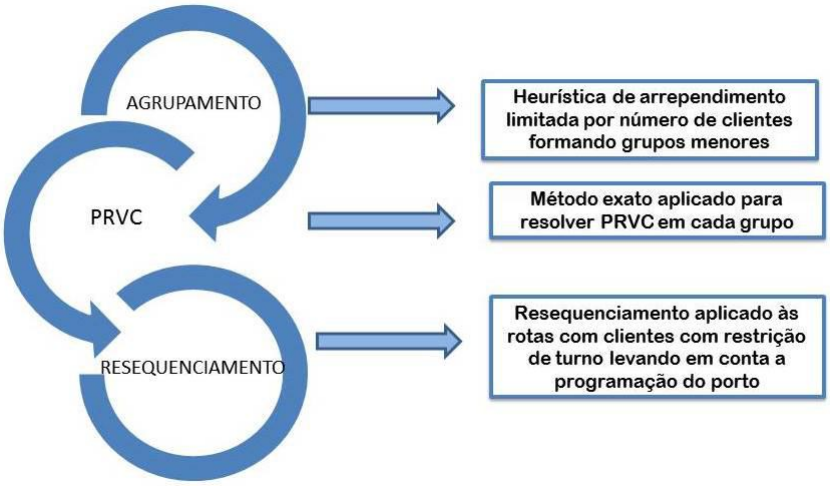


Figura 4.3: Etapas para resolver o PRESP por agrupamento-roteamento

baseada em rotas originais (rotas da empresa). Depois, o algoritmo tenta melhorar iterativamente a solução em uma heurística de dois passos.

O primeiro passo do algoritmo de agrupamento consiste em tentar melhorar a solução inicial, fazendo todas as trocas possíveis entre a posição de duas unidades. As soluções são avaliadas calculando a distância que seria necessária para visitar todas as unidades marítimas em cada grupo a partir do porto. Em cada iteração, a melhor solução é gravada e o processo reinicia com base nesta nova configuração. Para evitar ciclos, um procedimento de busca tabu também pode ser realizado, trazendo maior eficiência para o método. O processo se repete até que não seja encontrada uma solução melhor ou quando um número máximo de iterações é alcançado.

No segundo passo, o objetivo é aumentar o horizonte de possíveis soluções. Grupos são reconstruídos em torno de sementes, que são definidas como centro geométrico de cada grupo. Para atribuir cada unidade a grupos recém formados, uma função de arrependimento (4.28) é usada :

$$Regret = C_{ij^2} - C_{ij^1} \tag{4.28}$$

sendo j^1 a semente mais próxima da unidade i e j^2 a segunda semente mais próxima. Em outras palavras, esta função é definida como a penalidade por escolher a segunda semente mais próxima ao invés da primeira. Seguindo a ordem das prioridades definidas pela função de arrependimento, cada unidade é atribuída a um grupo, até que sua capacidade seja atingida. Se não for possível atribuir uma unidade ao seu grupo mais próximo, esta é atribuída a grupos cada vez mais distantes. Depois, o algoritmo retorna ao começo e é repetido até atingir um critério de parada. O processo é feito como mostra o Algoritmo 1 a seguir.

Algorithm 1 Heurística de agrupamento

```
1: Inicializa: s_best ← s ← solução inicial;
2: enquanto iter < iter_max faça
3:    enquanto Não encontra uma nova solução faça
4:       enquanto Todas as possíveis trocas não forem realizadas faça
5:          Mude a posição entre duas unidades na melhor solução
6:          se s < s_best e Lista tabu [s] = ∅ então
7:             s_best ← s
8:          fim se
9:       fim enquanto
10:   fim enquanto
11: fim enquanto
12: Calcula sementes para cada grupo na melhor solução
13: Calcula a função de arrependimento para cada unidade na melhor solução
14: Ordena as unidades de acordo com a função de arrependimento
15: Atribui unidades de acordo com a ordem anterior
16: retorna s_best;
```

4.2.2 Fase 2: Criar rotas resolvendo o PRVC com restrições de tempo total de viagem

Na Fase II, construímos rotas resolvendo o PRVC com restrições de tempo total de viagem dentro de cada grupo formado através da heurística de agrupamento explicada na Seção 4.2. Visto que a modelagem completa do PRESP foi descrita n Seção 4.1, podemos agora apenas referenciar quais conjuntos parâmetros e variáveis de decisão (Tabela 4.1) que são usadas nessa etapa, assim como as equações que são utilizadas para a construção inicial das rotas.

Notação	
Conjuntos:	
$\mathcal{V}$	Conjunto de vértices do grafo G
$\mathcal{C}$	Conjunto de clientes
$\mathcal{A}$	Conjunto de arcos do grafo G
$\mathcal{J}$	Conjunto de dias da semana
Parâmetros:	
s_i	Tempo de serviço no cliente i
z_j	Lingadas nos clientes i
k_j	Demanda no cliente i
ρ_j	Carga de retorno no cliente j
d_{ij}	Distância do nó i ao j
t_{ij}	Tempo de viagem do nó i ao j
R	Tempo máximo de viagem
δ	Capacidade máxima das embarcações
H	Tempo máximo de atracação no porto
α e β	Coeficientes da regressão usada pela companhia para encontrar o tempo de atracação no porto
Variáveis de Decisão:	
x_{ij}	Variável binária que leva valor 1 se o arco (i, j) está contido em alguma rota e zero caso contrário
f_{ij}	Fluxo de demanda que passa pelo arco (i, j)
g_{ij}	Fluxo de lingadas que passa pelo arco (i, j)
h_i^1	Número real que indica o horário de chegada no cliente i.

Tabela 4.1: Tabela contendo os parâmetros e variáveis de decisão usados na fase II

O conjunto de restrições utilizado para a resolução da fase II pode ser encontrado na subseção 4.1 e contém as seguintes referências: restrições de atribuição (4.2) e (4.3) com $i, j \in \mathcal{C}$; restrições de fluxo no porto (4.6), fluxo de demanda (4.10) e lingadas (4.11); restrições limitantes de capacidade da embarcação (4.7), de lingadas (4.8) e garantia de área disponível para a primeira carga de retorno (4.9) com $i, j \in \mathcal{C}$; restrições (4.15) de horários de chegada em cada cliente; restrições de tempo máximo de viagem (4.16); e restrições de tempo máximo de atracação (4.17).

As restrições (4.4), (4.5), (4.12), (4.13), (4.14) e as de (4.18) a (4.25) não são utilizadas nessa fase pois são especificas do problema com horários de abertura e tempo entre viagens. Finalmente, as restrições (4.24) e (4.25) que aplicam as condições de integralidade e não-negatividade nas variáveis reais binárias e não-negativas, respectivamente são substituídas por:

$$x_{ij} \in \{0, 1\} \qquad i, j \in \mathcal{V} \quad (4.29)$$

$$h_i, f_{ij}, g_{ij} \geq 0 \qquad i, j \in \mathcal{V} \quad (4.30)$$

4.2.3 Fase 3: Resequenciamento de rotas resolvendo o PRESP

Após o conhecimento da programação do porto contendo os horários de partida para as possíveis rotas geradas na fase II, a fase III é realizada. Agora, o modelo leva em consideração as novas restrições de tempo: horários de abertura, tempos de ciclo e horários de partida do porto. O mesmo conjunto de unidades é mantido em cada rota de atendimento prevista para primeira e segunda viagem, no entanto, com a possibilidade de mudanças na ordem das visitas devido às restrições de horário de abertura. O objetivo agora é minimizar os tempos de espera antes de atender clientes que abrem apenas de dia ou de noite e manter o tempo de ciclo para cada unidade em torno de 3,5 dias (2 vezes por semana). Esta etapa é aplicada apenas em rotas em que pelo menos uma instalação tenha restrições de horário de abertura porque o cronograma do porto já respeita o tempo de ciclo para as rotas que permaneceram com a mesma sequencia de primeira e segunda viagem.

Para refazer a sequência da rota, o método exato para a fase III também faz o uso de um modelo de fluxo, mas com conjuntos, parâmetros e algumas restrições que podem ser vistos na Subseção 4.1. A Tabela 4.1 e a Tabela 4.2 apresentam todas as notações usadas nessa fase.

Notação	
Conjuntos:	
$\mathcal{P}$	Conjunto de clientes pertencentes a primeira viagem
$\mathcal{B}$	Conjunto de clientes pertencentes a segunda viagem
$\mathcal{T}$	Conjunto de clientes com restrição de turno
$\mathcal{D}$	Conjunto de clientes com restrição de turno diário
$\mathcal{N}$	Conjunto de clientes com restrição de turno noturno
$\mathcal{S}$	Conjunto de todos os nós auxiliares
$\mathcal{S}^1$	Conjunto de nós auxiliares correspondentes a primeira viagem
$\mathcal{S}^2$	Conjunto de nós auxiliares correspondentes a segunda viagem
$\mathcal{J}$	Conjunto de dias da semana
Parâmetros:	
$[a_i^d, b_i^d]$	Horas abertura de clientes $i \in \mathcal{T}$
γ_i^d	Horas de partida do porto no nó auxiliar i no dia d
$[n_{min}, n_{max}]$	Intervalo de tempo de ciclo
μ	Peso que varia de acordo com a magnitude da soma ds distâncias
Variáveis de decisão:	
y_i^d	Variável binária recebe valor um se o dia d é escolhido para servir o cliente $i \in \mathcal{T}$ e zero caso contrário
w_i	Tempo de espera antes do atendimento no cliente $i \in \mathcal{T}$

Tabela 4.2: Tabela de parâmetros e variáveis de decisão na fase III

Como na fase II, o conjunto de restrições utilizado para a resolução da fase III pode ser encontrado na subseção 4.1. Entretanto esse terceiro modelo não utiliza algumas restrições que são específicas quando se trata mais de uma rota por grupo, o que não é o caso já que re-sequenciamos somente rotas que possuem clientes com

horário de abertura separadamente. As restrições que precisam ser modificadas ou são desnecessárias a esse modelo implementado na fase III são: restrições (4.12), (4.13) e (4.15) são descartadas e as restrições de tempo máximo de viagem (4.16) são substituídas por (4.31) que incluem o horário de saída do porto:

$$h_j^1 + s_j + t_{j0} - h_i - M(1 - x_{j0}) \leq R \qquad i \in \mathcal{S}, j \in \mathcal{C} \quad (4.31)$$

4.3 Heurística ALNS

Adaptive large neighborhood search é uma estrutura de busca local em que uma grande coleção de variáveis é modificada em cada iteração através de alguns operadores de remoção e inserção simples e rápidos selecionados de forma independente. Em cada iteração são escolhidos operadores para destruir parcialmente a solução atual e para reparar essa mesma solução. Em seguida a nova solução é aceita se satisfizer alguns critérios definidos.

Alternativamente, podemos ver essa heurística adaptável como uma sequência de operações de correção e otimização em que um número de variáveis são corrigidas no seu valor atual e a operação de otimização procura encontrar uma solução quase ótima que respeite as variáveis reparadas, finalizando com todas as variáveis desbloqueadas novamente. Este é um ponto de vista interessante para os problemas nos quais as operações de destruição e reparação não parecem intuitivas [7].

O ALNS é uma extensão do LNS apresentado por SHAW [39] que modifica uma grande coleção de variáveis em cada iteração e também se baseia no paradigma *Ruin e Recreate* apresentado por SCHRIMPF *et al.* [60], ou o *Ripup and Reroute* paradigma [61]. A diferença entre LNS e ALNS está na seleção dos operadores. Enquanto no LNS os operadores são selecionados com probabilidades iguais, no ALNS existe um mecanismo adaptativo probabilístico dependente do desempenho anterior.

O ALNS também tem semelhanças com *very large neighborhoods search* (VLNS) apresentado por AHUJA *et al.* [62], que é um algoritmo que opera sobre vizinhanças muito grandes escolhidas de forma a que ainda possam ser pesquisadas de forma eficiente.

A heurística VNS foi apresentada por HANSEN e MLADENOVIĆ [63] e usa um conjunto parametrizado de vizinhanças normalmente obtidas usando busca em profundidade em uma dada vizinhança. Quando o algoritmo encontra um mínimo local ele continua nessa grande vizinhança do conjunto parametrizado e ao sair do mínimo local o algoritmo continua numa vizinhança menor. Ao contrário disso, o ALNS opera sobre um grande conjunto de vizinhanças predefinido correspondente às

heurísticas de remoção e inserção. A diferença entre os dois é ilustrada graficamente por PISINGER e ROPKE [7] e reproduzida na Figura 4.4. Nas subseções que seguem são mostrados os mecanismos utilizados na construção da heurística ALNS.

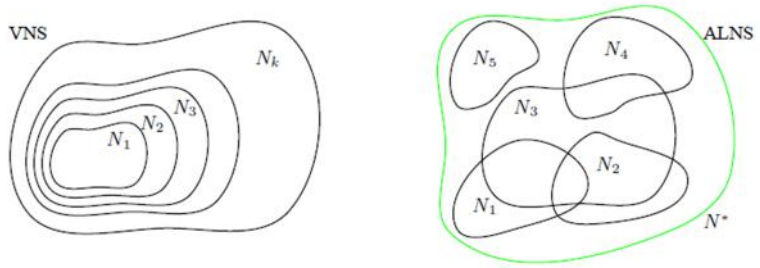


Figura 4.4: Diferença estrutural entre as vizinhanças N utilizadas pela heurística VNS e ALNS. VNS opera com uma vinhança e busca em profundidade enquanto no ALNS as vizinhanças definidas pelas heurísticas de busca.

4.3.1 Mecanismo adaptativo

O mecanismo adaptativo é destinado a escolha dos operadores de remoção e inserção levando em conta seus resultados anteriores e para isso um peso é associado a cada operador. Supondo que temos h operadores de remoção ou inserção, cada um com um peso w_j, $j = 1, ..., h$. O operador de remoção ou inserção i é então selecionado com probabilidade

$$\frac{w_i}{\sum_{i=1}^{h} w_j}, i = 1, ...h. \tag{4.32}$$

Isto significa que a probabilidade de selecionar um dado operador aumenta com seu peso. Começando com um peso unitário para cada operador de remoção e inserção, os pesos são atualizados após um número consecutivo de iterações, chamado de segmento. Os pesos de um dado segmento φ são baseados naqueles usados no segmento anterior $\varphi - 1$ e são calculados da seguinte forma:

$$w_i^{\varphi} = \gamma.w_i^{\varphi-1} + (1 - \gamma).\pi_i^{\varphi-1} \tag{4.33}$$

em que γ possui valor no intervalo (0,1), $\pi_i^{\varphi-1}$ é a pontuação do operador i no final do segmento $\varphi - 1$. Esta pontuação, zerada no começo de cada segmento, é aumentada quando o operador de inserção ou destruição i é usado numa dada iteração t para produzir uma nova solução. Mais precisamente a nova pontuação na iteração $t + 1$ se torna

$$\pi_i^{t+1} = \pi_i^t + \begin{cases} \pi_1, & \text{se encontrou uma nova melhor solução;} \\ \pi_2, & \text{se a solução encontrada é melhor que a solução atual;} \\ \pi_3, & \text{se a solução encontrada não for melhor,} \\ & \text{mas ainda for aceita com uma dada probabilidade.} \end{cases}$$

em que π_1, π_2, π_3 são parâmetros.

O chamado fator de reação γ controla a inércia na equação de atualização dos pesos. Quando φ assume valores perto de 1, o histórico prevalece e os pesos não mudam muito. Por outro lado quando φ assume valores perto de zero, a atualização é dirigida pelo peso mais recente. Na implementação da heurística ALNS pra o PRESP escolhemos $\pi_1 = 50, \pi_2 = 20, \pi_3 = 5$ como sugerido em RIBEIRO e LAPORTE [44].

4.3.2 Critérios de aceitação e parada

Os critérios de aceitação são baseados no mesmo critério comumente usado no recozimento simulado (SA) que foi proposto por METROPOLIS *et al.* [64] e consiste em avaliar a diferença de energia entre a solução atual e a nova solução calculando $\Delta = E_{k+1} - E_k$. Se Δ é negativo, isto significa que a solução atual é melhor que a anterior, e esta última é substituída. Se Δ é positivo, a probabilidade de esta solução de maior energia substituir a anterior é dada por $p = \exp(-\Delta/T)$. Transições muito ruins (ou seja, com Δ grande) são menos prováveis que as transições "pouco ruins". Quanto menor a temperatura, menos provável é que uma transição ruim seja aceita. O comportamento da função probabilidade pode ser visto na Figura 4.5. Observe que para um valor de Δ/T igual a 5 (Δ 5 vezes maior que a temperatura) a probabilidade de aceitação é praticamente nula, e que a probabilidade de aceitação de 50% corresponde a $\Delta/T = ln(2) = 0.69$.

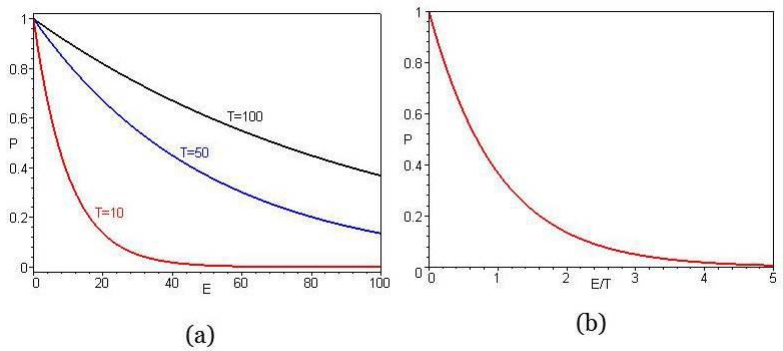


Figura 4.5: Probabilidade de aceitação de uma transição de maior energia:(a) $p \times \Delta$ para vários valores de temperatura T; (b) $p \times \Delta/T$

O método começa sorteando um número aleatório r a partir de uma distribuição uniforme no intervalo $[0, 1]$. Considere que p representa a probabilidade da transição

para este novo ponto ser aceita. A probabilidade do número sorteado $r \leq p$, também contido no intervalo $[0, 1]$, é (100p)%. Logo, se $r < p$, a transição é aceita. Do contrário, este ponto é descartado e um novo vizinho do ponto atual é calculado.

Os valores inicial e final da temperatura dependem da escala da função. Lembrando que há 50% de chance de aceitar uma transição para uma solução pior quando $\Delta/T = 0.69$. Alguns trabalhos na literatura sugerem que sejam definidos os parâmetros de resfriamento tais que 80% das transições pobres sejam aceitas no início do algoritmo. No entanto, esse critério é difícil de ser aplicado na prática, e nem sempre corresponde à melhor escolha de parâmetros.

Descrevendo formalmente, sejam $z(s)$ o custo da solução e $T > 0$ a temperatura. Dada uma solução atual s, uma solução em avaliação s' sempre é aceita se $z(s') < z(s)$, e é aceita com probabilidade $e^{-(z(s')-z(s))/T}$ caso contrário. A temperatura começa em T_0 e diminui pelo fator de taxa de resfriamento γ em cada iteração, com $0 < \gamma < 1$. O critério de parada é satisfeito quando $T < T_F$.

A fim de que a temperatura de congelamento (T_F) e o número máximo de iterações ($iter_{max}$) sejam ambos critérios de parada, o valor da taxa de resfriamento (α_T) foi escolhido considerando o valor das temperaturas ao longo das iterações, visto que a temperatura é modificada a cada iteração ($iter$). Para se chegar à Equação (4.34) é necessário um desenvolvimento algébrico com K de iterações e as relações iniciais ($iter = 0 \Rightarrow T = T_0$) e finais ($iter = k \Rightarrow T = T_F$). Desta maneira α_T deixa de ser um parâmetro e passa a ser dependente de T_0, T_F e K.

$$\alpha_T = \sqrt[k-1]{\frac{T_F}{T_0}} \tag{4.34}$$

A escolha da temperatura inicial T_0 para esse trabalho foi feita através de calibração desse parâmetro e pode ser encontrada no Capítulo 5.

4.3.3 Solução inicial

A heurística ALNS é inicializada com uma solução obtida através do algoritmo de varredura (Algoritmo 2) proposto em GILLETT e MILLER [65]. Ele é usado para PRVC com um ou mais depósitos e nós localizados em um plano Euclidiano. Para facilitar a implementação, as UMs e o porto são representados por suas coordenadas polares (θ_i, ρ_i) atribuindo o valor $\theta_i = 0$ ao nó que representa o porto $(i \leftarrow p)$ e os ângulos remanescentes de (i, p) são calculados. Depois disso, as UMs são ordenadas em ordem ascendente de seus ângulos. A implementação para o PRESP ocorre da seguinte forma: primeiro escolhe-se uma embarcação que não tenha sido usada e, em seguida, atribui-se as UMs a embarcação a partir de uma UM não roteada com o menor ângulo, desde que sua capacidade (K) e tempo de viagem total (T_t) não sejam excedidos. Os turnos não são tratados como restrição e sim como uma penalidade

na função objetivo, ou seja o inicio do atendimento a essas unidades com turnos de funcionamento diurno ou noturno é penalizados com um tempo de espera até que a janela de tempo de atendimento se abra. O algoritmo finaliza quando não existem mais UMs a serem roteadas, como representado na Figura 4.6.

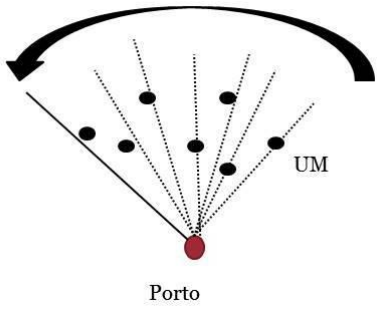


Figura 4.6: Visualização do funcionamento do algoritmo de varredura com o porto como ponto de partida e roteamento das UMs.

Algorithm 2 Algoritmo de varredura

1: Inicializa: $K \leftarrow$ capacidade total, $T_t \leftarrow$ Tempo total;
2: Converta os vértices i de coordenadas geográficas para polares (θ_i, ρ_i);
3: Atribua um vértice inicial $i \leftarrow p$;
4: Calcule os ângulos restantes partindo de $(1, p)$;
5: Crie uma lista com os vértices ordenados de forma crescente do ângulo;
6: **enquanto** Existir um vértice i não agrupado **faça**
7: Escolha um veículo j não utilizado;
8: **enquanto** $\sum_i demanda_i < K_j$ e $\sum_i tempo_i < T_t$ **faça**
9: Acrescente um vértice i a rota atendida pelo veículo K;
10: **fim enquanto**
11: **fim enquanto**
12: **retorna** solução s

4.3.4 Operadores de remoção e inserção

Depois de encontrar uma solução inicial, excluímos e inserimos UMs com os operadores de destruição e reparação. Para determinar o número de UMs que são removidas em algum percurso, criamos uma função. Essa quantidade é escolhida de forma aleatória no intervalo $[1, n]$, sendo n um número aleatório com distribuição semi triangular com uma inclinação negativa. Como mostrado por PISINGER e ROPKE [7], há uma grande transformação da solução quando n possui um valor alto, então, é necessário escolher mais vezes uma quantidade pequena de retiradas n. Devido a uma grande quantidade de combinações para a inserção, se n e o número total de clientes forem grandes, o tempo computacional também o será.

Como podemos ver no exemplo da Figura 4.7, a solução $s = \{\{0, 1, 2, 0\}, \{0, 3, 4, 5, 0\}\}$ foi destruída com a retirada do cliente 3 e reparada gerando a solução $s' = \{\{0, 1, 2, 3, 0\}, \{0, 4, 5, 0\}\}$.

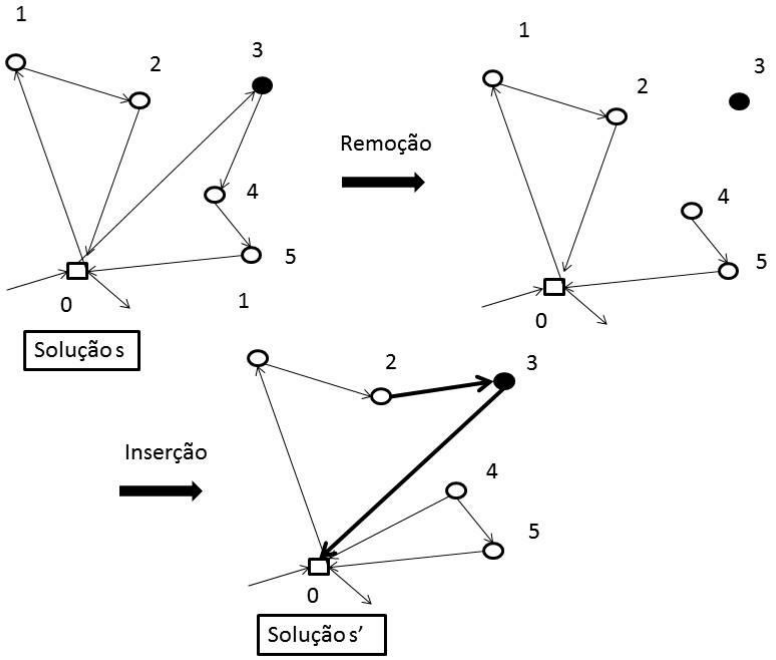


Figura 4.7: Visualização de destruição e reparação de uma solução no ALNS

Os operadores de remoção escolhidos foram os seguintes:

1. *Remover aleatoriamente n clientes*
   Esta heurística de remoção remove aleatoriamente a quantidade n de clientes das primeiras e segundas viagens na solução atual s, gerando a solução em avaliação s', como é mostrado no Algoritmo 3. A característica principal desse operador é diversificar a pesquisa quando se está preso em um mínimo local.

Algorithm 3 Remoção aleatória

Inicializa: s
$n \leftarrow$ quantidade de clientes a serem removidos;
enquanto $iter < n$ **faça**
  Escolhe uma rota r aleatoriamente da primeira e segunda viagens;
  Escolhe um cliente $i \in r$ aleatoriamente;
  Remove o cliente i da primeira e segunda viagens;
  $iter \leftarrow iter + 1$;
fim enquanto
$s' \leftarrow s$
retorna s'

2. *Remover n clientes com maior economia de distância*

   Este operador identifica qual cliente representa a maior distância total percorrida pelo veículo se ele for deixado na rota e o remove. O processo continua até que n clientes sejam retirados. Esta heurística visa reduzir tempo computacional quando calcula apenas uma parte da função objetivo. Veja o Algoritmo 4 para mais detalhes.

Algorithm 4 Remoção maior economia de distância

```
1:  Inicializa: s
    n ← quantidade de clientes a serem removidos;
2:  enquanto iter < n faça
3:      para Cada rota r da primeira viagem faça
4:          para Cada cliente i ∈ r faça
5:              Remove i;
6:              Calcula distância total;
7:              se Distância total atual < Distância total anterior então
8:                  d ← i;
9:              fim se
10:             Reinsere i;
11:         fim para
12:     fim para
13:     Remove d da primeira e segunda viagens;
14:     iter ← iter + 1;
15: fim enquanto
16: retorna s′
```

3. *Remover n clientes com a maior economia de tempo*

   Este operador remove os clientes que representam a maior economia de tempo quando não são considerados na solução (Algoritmo 5). Os tempos de espera antes do serviço nos clientes devem ser minimizados na função objetivo.

Algorithm 5 Remoção maior economia de tempo

```
1:  Inicializa: s
    n ← quantidade de clientes a serem removidos;
2:  enquanto iter < n faça
3:      para Cada rota r da primeira viagem faça
4:          para Cada cliente i ∈ r faça
5:              Remove i;
6:              Calcula tempo total;
7:              se Tempo total atual < Tempo total anterior então
8:                  d ← i;
9:              fim se
10:             Reinsere i;
11:         fim para
12:     fim para
13:     Remove d da primeira e segunda viagens;
14:     iter ← iter + 1;
15: fim enquanto
16: retorna s′
```

4. *Remover n clientes com o custo total menor*

   Este operador representado no Algoritmo 6 remove o cliente que representa o menor valor da função objetivo quando não são considerados na solução. Apesar de ser o algoritmo mais completo, é preciso mais tempo computacional para resolver o problema porque considera todas as variáveis de tempo e distância.

Algorithm 6 Remoção cliente com menor fo

```
1: Inicializa: s
   n ← quantidade de clientes a serem removidos;
2: enquanto iter < n faça
3:   para Cada rota r da primeira viagem faça
4:     para Cada cliente i ∈ r faça
5:       Remove i;
6:       Calcula fo;
7:       se fo atual < fo anterior então
8:         d ← i;
9:       fim se
10:      Reinsere i;
11:    fim para
12:  fim para
13:  Remove d da primeira e segunda viagens;
14:  iter ← iter + 1;
15: fim enquanto
16: retorna s'
```

5. *Remover todos os clientes de uma rota*

   Como dito anteriormente a quantidade de clientes retirados por iteração não deve ser alta devido à grande quantidade de inserções possíveis, gerando um maior tempo computacional. Entretanto esse operador de remoção de rotas completas pode ser utilizado de maneira que ajude a diversificar as soluções. Como pode ser visto no Algoritmo 7, a remoção da rota é uma heurística na qual tentamos remover grupos de clientes relacionados a alguma rota. Ao remover um cliente de uma rota, há uma grande possibilidade de que os operadores de inserção o re-aloquem na mesma rota, por isso removendo uma rota completa os operadores de inserção são obrigados a optar por uma rota diferente.

Logo depois de usar os operadores de destruição, o mesmo número de clientes removidos da solução acima é reinserido de acordo com o operador de reparo escolhido. É importante enfatizar que cada cliente é inserido em diferentes posições, mas na mesma rota para s duas viagens. E os critérios de viabilidade são a capacidade máxima do veículo, tempo de viagem total e atracação permitidos.

1. *Inserção aleatória de n clientes*

   Os clientes, antes removidos, agora são reinseridos em rotas e posições

Algorithm 7 Remoção completa da rota

1: Inicializa: s
   $n \leftarrow$ quantidade de clientes a serem removidos;
2: Escolha uma rota r aleatoriamente da primeira e segunda viagens;
3: Seja L o conjunto de clientes pertencentes a rota r a serem retirados;
4: **enquanto** $L \neq \emptyset$ **faça**
5: Escolha um cliente i;
6: Remove o cliente i da primeira e segunda viagens da rota;
7: **fim enquanto**
8: **retorna** s'

aleatórias ad solução em avaliação s' que asseguram a viabilidade como visto no Algoritmo 8. Semelhante ao operador de remoção aleatória, sua função é diversificar a pesquisa quando está presa em um mínimo local.

Algorithm 8 Inserção aleatória

1: Inicializa: s'
   I o conjunto de clientes a serem inseridos;
2: **para** $i \in I$ **faça**
3: Escolha uma posição n da rota r aleatoriamente para inserir;
4: **se** Não excede a capacidade do navio, o tempo máximo de viagem e atracação **então**
5: Insere o cliente i na posição n da rota r;
6: **fim se**
7: **fim para**
8: **retorna** s'

2. *Melhor inserção*

   Este operador avalia um a um e insere os clientes de acordo com o menor valor da função objetivo. Este processo é repetido até que todos os clientes removidos tenham sido inseridos. O Algoritmo 9 descreve com detalhes a escolha da melhor inserção a partir da solução em avaliação s'.

3. *2-regret*

   O último operador de inserção é implementado visando suavizar o comportamento míope da heurística de menor custo. Isso é feito definindo uma ordem de reinserção baseada em medidas de arrependimento descrito no Algoritmo 10. Para um determinado cliente esta heurística calcula um valor de arrependimento igual à diferença entre duas soluções nas quais o cliente é inserido na melhor rota ou na segunda melhor rota, respectivamente. O cliente com o maior valor de arrependimento é escolhido para ser inserido na solução em avaliação s'. Este operador tenta melhorar a solução incorporando um tipo de informação futura ao selecionar a solicitação para inserir. Formalmente seja $\mathcal{C}$

Algorithm 9 Melhor inserção

```
Inicializa: s'
I o conjunto de clientes a serem inseridos f* = ∞
para iter < |I| faça
  para i ∈ I faça
    para Cada rota r faça
      para Cada posição n da rota r faça
        se Não excede a capacidade do navio, o tempo máximo de viagem e atracação então
          Insere o cliente i na posição n da rota r
          se f(s') < f* então
            n* ← n
            f* = f(s')
          fim se
        fim se
      fim para
    fim para
  fim para
  Insere o cliente i na posição n* da solução s';
  iter ← iter + 1;
fim para
retorna s'
```

o conjunto de clientes e $x_{ik} \in R$ a variável que indica a rota para cada cliente $i \in \mathcal{C}$ com k^{esimo} menor custo de inserção, e $\Delta f_{i,x_{ik}} \leq \Delta f_{i,x_{ik'}}$ para $k \leq k'$. Em cada iteração a heurística escolhe inserir um cliente i que maximize

$$\max_{i \in \mathcal{C}} \{\Delta f_{i,x_{i2}} - \Delta f_{i,x_{i1}}\}. \tag{4.35}$$

O pedido é inserido em sua posição de custo mínimo. Informalmente falando, escolhemos a inserção que nos arrependeremos mais se não for efetuada imediatamente.

4.3.5 Algoritmo ALNS

O Algoritmo 11 mostra o pseudocódigo da heurística ALNS. O algoritmo inicia com todos os pesos e pontuações utilizados para avaliar os operadores de remoção e inserção. Além disso, é atribuída uma solução inicial encontrada pelo algoritmo de varredura às variáveis que representam as soluções atuais s e as melhores soluções encontradas s_{best}. Os parâmetros de temperatura inicial T_0 e final T_F são escolhidos. Enquanto o número máximo de iterações i_{max} não for atingido, são aplicados os métodos de remoção e inserção de clientes a s gerando uma solução em avaliação s'. Em seguida a solução s' é analisada de maneira que se ela for melhor que a

Algorithm 10 2-Regret

```
1: Inicializa: s'
   I o conjunto de clientes a serem inseridos
   F lista de valores da função objetivo f(s')
   f* ← ∞
2: para iter < |I| faça
3:   para i ∈ I faça
4:     para Cada rota r ∈ s' faça
5:       para Cada posição n da rota r faça
6:         se Não excede a capacidade do navio, o tempo máximo de viagem e
           atracação então
7:           Insere cliente i na posição n da rota r
8:           se f(s') < f* então
9:             n* = n
10:            F ← f(s')
11:            f* = f(s')
12:            Ordena F
13:            Calcula 2-regret
14:          fim se
15:        fim se
16:      fim para
17:    fim para
18:  fim para
19:  Insere o cliente i com maior arrependimento na posição n* da rota r da solução
     s';
20:  iter ← iter + 1;
21: fim para
22: retorna s'
```

melhor solução encontrada os operadores recebem pontuação π_1; se for melhor que a solução atual é atribuída a pontuação π_2 e se for aceita com o critério utilizado na heurística SA a pontuação será π_3. Como visto anteriormente na Subseção 4.3.1 essas pontuações são dadas de acordo com a qualidade da solução gerada pelo operador e a cada φ iterações essas pontuações e os pesos são atualizados e as pontuações são zeradas. Por fim a temperatura T para cada iteração é atualizada e o contador de iterações i é acrescido de uma unidade. O algoritmo retorna a melhor solução encontrada.

4.4 Heurística híbrida ALNS-CS

Muitos trabalhos recentes sobre metaheuristica híbrida referem-se a metaheurísticas que empregam heurísticas de busca local específicas para o problema tratado, objetivando acelerar o processo de busca dessas metaheurísticas. Heurísticas de busca local possuem um custo computacional muito alto, pois em geral necessitam recalcu-

Algorithm 11 Heurística ALNS

```
1: Inicializa: definir todos os pesos igual a 1 e todas as pontuações igual a zero.
   s_best ← s ← solução inicial, T ← T_0.
2: enquanto i < i_max faça
3:    s' ← s;
      Aplicar os operadores de destruição e reparação à s' e atualizar o número de vezes que eles
      foram usados.
4:    se z(s') < z(s) então
5:       s ← s';
6:       se z(s) < z(s_best) então
7:          s_best ← s;
            atualize a pontuação para os operadores usados com π_1;
8:       senão
9:          atualize a pontuação dos operadores usados com π_2;
10:      fim se
11:   senão
12:      se s' é aceita pelo critério do Simulated Anneling então
13:         s ← s';
            atualize a pontuação para os operadores usados com π_3;
14:      fim se
15:   fim se
16:   se contador de iterações é múltiplo de φ então
17:      atualize os pesos de todos os operadores e zere as pontuações.
18:   fim se
19:   T ← α_T * T;
20:   i ← i + 1;
21: fim enquanto
22: retorna s_best;
```

lar o valor da função objetivo a cada solução pesquisada. Sendo assim sua utilização indiscriminada pode levar a um crescimento significativo da complexidade dos algoritmos.

A heurística ALNS, diferentemente de outras, utiliza-se de diferentes operadores de remoção e inserção para encontrar novas soluções. Esse fato faz com que a função objetivo não seja calculada por completo repetidamente, reduzindo o tempo computacional. Como foi visto anteriormente, a heurística ALNS escolhe seus operadores de forma aleatória e atribui uma pontuação para que ao final, já com a probabilidade de aceitação baixa, possam ser utilizados somente os operadores com pontuação mais alta e que em geral são os que trabalham com a função objetivo completa.

O aperfeiçoamento proposto neste trabalho para a heurística ALNS é a aplicação de busca local somente em regiões consideradas promissoras ao invés de ter como direcionador principal a avaliação de operadores. Uma maneira de detectar essas regiões é por meio da quantidade de soluções geradas em uma dada região do espaço de busca. O problema passa a ser então como identificar estas regiões promissoras. A hibridização com o CS visa, portanto, responder a esta necessidade e pode trazer diversos benefícios ao intensificar a busca apenas em regiões supostamente promissoras, tais como, redução do número de buscas locais sem prejudicar o desempenho

do algoritmo, pesquisa das principais regiões do espaço de busca ao longo de todo o processo, e aplicação de busca local em soluções com boa qualidade que necessitam de poucas alterações para se tornarem ainda melhor.

Considerando que o ALNS e as metaheurísticas em sua maioria privilegiam as soluções de melhor qualidade e estas influenciam no processo de busca, é esperado que se tenha uma maior amostragem de soluções próximas as regiões destas soluções. Sendo assim é possível estabelecer as regiões nas quais muitas soluções são geradas e tendem a ser regiões promissoras, isto é, regiões que contenham as melhorem soluções durante toda a busca.

Os detalhes da aplicação da heurística híbrida ALNS-CS são agora descritos, esclarecendo a abordagem. Essa nova heurística pode ser vista como uma técnica para o aperfeiçoamento da heurística ALNS através da aplicação de algumas técnicas desenvolvidas no CS.

4.4.1 Definição formal do CS

O método de pesquisa em grupos de soluções (CS) é um método híbrido que combina adequadamente metaheurísticas e heurísticas de buscas locais. O CS tenta localizar áreas de pesquisa promissoras enquadrando-as por grupos. A finalidade dessa metodologia é refinar o processo de escolha de soluções antes de aplicar a busca local em vez de escolher aleatoriamente ou aplicar busca local em todas as soluções geradas por uma metaheurística. Portanto, espera-se uma melhora no processo de convergência associado a uma diminuição do esforço computacional devido ao uso mais racional de métodos de busca locais.

O algoritmo CS generaliza o ECS proposto por OLIVEIRA e LORENA [9, 48] e substitui o algoritmo evolutivo por metaheurísticas como SA, GRASP ou tabu search. O ECS emprega um processo de agrupamento para detectar áreas promissoras no espaço de busca executando-o simultaneamente com um algoritmo evolutivo e identificando grupos de indivíduos que merecem atenção especial. É particularmente interessante descobrir essas áreas o mais rápido possível para mudar a estratégia de pesquisa sobre elas.

Componentes do CS e suas atribuições

CS pode ser dividido em quatro partes conceitualmente independentes com diferentes atribuições:

- Metaheurística de busca (MB) ,
- Agrupamento iterativo (AI),
- Módulo de análise (MA) e

- Módulo de busca local (BL) .

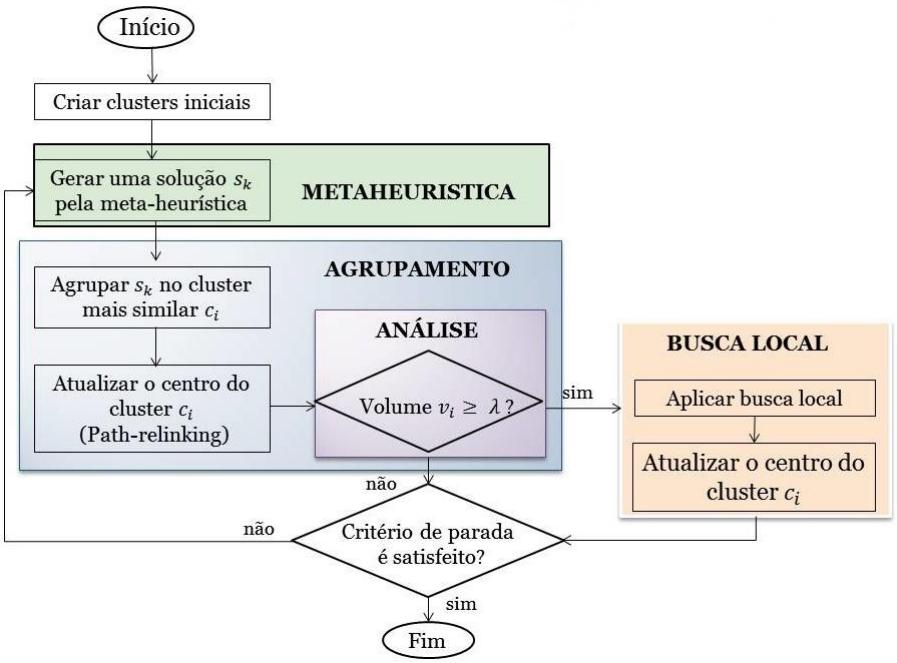


Figura 4.8: Fluxograma da heurística CS

O MB é composto por um algoritmo de otimização independente que funciona como um gerador de soluções em tempo integral a fim de obter soluções diversificadas no espaço de busca. O objetivo do AI é reunir soluções similares em grupos, mantendo um centro comum representativo para eles. Esse componente é, portanto, importante processar desde que tenha uma função classificadora, mantendo apenas informações relevantes agrupadas e aumentando a intensificação da busca nas áreas de busca promissoras. Uma métrica de distância deve ser definida, a priori, permitindo uma medida de similaridade para o processo de agrupamento.

O MA fornece uma análise de cada grupo de soluções, em intervalos regulares de geração, indicando um provável grupo promissor. Normalmente, a densidade do grupo é usada nesta análise, ou seja, o número de seleções ou atualizações que aconteceram recentemente. O MA também é responsável pela eliminação dos grupos com densidades mais baixas.

Por fim, o BL é um módulo de busca interno que fornece a exploração de uma suposta área de busca promissora, emoldurada por aglomerados. Esse processo pode acontecer depois de ter descoberto um grupo alvo ou pode ser um processo contínuo, inerente ao AI, sendo executado sempre que uma nova solução é agrupada. O fluxograma da Figura 4.8 trás seu projeto conceitual.

O processo de agrupamento de soluções

O espaço de busca precisa ser dividido em regiões promissoras visando formar grupos de soluções. O número de grupos NC pode ser fixado a priori [49] ou dinamicamente determinado de acordo com o tamanho a área de busca a ser explorada pela heurística [9]. Neste último caso, os grupos podem ser criados de forma que todas as soluções candidatas sejam cobertas, pelo menos, por um grupo. Por outro lado, grupos inativos, ou seja, que não cobrem nenhuma área de soluções, podem ser eliminados.

A cobertura do espaço de soluções é determinada por uma métrica de distância que calcula a semelhança entre uma determinada solução e o centro do grupo e deve considerar a natureza do problema. Por exemplo, em otimização contínua sem restrições, a semelhança foi definida em relação à distância Euclidiana [9]. Na otimização combinatória, a similaridade pode ser definida como o número de movimentos necessários para deslocar uma solução para o centro do grupo [48]. Em geral, cada grupo i é definido com três atributos $G_i = (c, v, r)$ onde c_i, v_i e r_i são o centro, o volume e o índice de ineficácia do grupo i.

Inicialmente, o centro c_i é obtido aleatoriamente e, progressivamente, tende a se direcionar a pontos realmente promissores no subespaço próximo. O volume total do grupo v_i pode ser calculado, considerando a natureza do problema. É importante definir um subespaço de pesquisa adequado para ser explorado pelas estratégias de pesquisa associadas ao grupo. A estratégia de busca é uma intensificação de busca sistemática, em que as soluções de um grupo interagem entre si ao longo do processo de agrupamento, gerando novas soluções. O índice de ineficácia r_i é uma variável de controle para identificar se a busca local está ou não melhorando o centro do grupo G_i. O valor de r_i indica o número de vezes consecutivas que a busca local foi aplicada ao grupo G_i e não melhorou a solução. Este atributo evita que buscas locais sejam executadas por mais de r_{max} vezes em regiões ruins ou regiões que já tenham sido exploradas.

O CS inicia seu processo criando grupos de soluções iniciais aleatórios, e a cada iteração uma solução s_k é gerada pela metaheurística e enviada para o processo de agrupamento. Então s_k é inserido no grupo mais semelhante, ou seja, aquele que miniminiza a distância métrica escolhida entre s_k e o centro do grupo c_i. Caso contrário, se a informação for considerada suficientemente nova (distância métrica suficientemente grande), ela é considerada como um centro em um novo grupo de soluções. Um método de assimilação geralmente é utilizado para aplicar estratégias de intensificação e diversificação dentro dos grupos de soluções. O centro do grupo é atualizado com informações contidas na nova solução agrupada por meio do processo de assimilação, fazendo com que o centro se desloque no espaço de busca.

Para examinar cada grupo de soluções, a heurística usa o componente MA, indicando um provável conjunto promissor. O volume do grupo v_i é uma medida que indica o nível de atividade do grupo, ou seja, conta o número de soluções geradas pelo MB e agrupadas em G_i. Quando v_i atinge um certo limite λ definido a priori, significa que algum modelo de informação se torna predominantemente gerado por MB e um módulo BL é aplicado no centro c_i intensificando uma pesquisa na vizinhança do grupo. No entanto, se o método BL não tiver sido bem sucedido nas últimas r_{max} aplicações neste grupo promissor, uma perturbação aleatória é aplicada no centro c_i, com o objetivo de escapar dessa região do espaço de busca.

A busca obtém sucesso em um grupo quando encontra uma solução que seja a melhor obtida até o momento. Encerrado o processo de agrupamento, retorna-se para a metaheurística que irá gerar outra solução. O pseudo-código do CS pode ser encontrado em Algoritmo 12.

Algorithm 12 Clustering search

```
Gerar soluções iniciais aleatórias
Cria um grupo ou mais de soluções
Escolhe uma solução inicial
enquanto Condição de término não é satisfeita faça
    Gera solução s_k por metaheuristica
    Procura o grupo com centro c_i mais similar à solução s_k
    Insere s_k dentro do grupo c_i (v_i ← v_i + 1)
    c_i ← Assimilação (c_i, s_k)
    se v_i ≥ λ então
        v_i ← 0
        Procura a melhor vizinhança de c_i
    fim se
    Atualiza o centro do grupo c_i
fim enquanto
retorna s_best
```

O critério de parada do CS pode ser definido pelo número máximo de iterações do processo de agrupamento ou pelo critério de parada da metaheurística escolhida.

4.4.2 Visão geral da heurística híbrida

Neste trabalho são propostas modificações para a heurística ALNS objetivando principalmente melhorar o método com a inserção de técnicas do CS. Uma vez que as soluções geradas pela heurística ALNS não são investigadas por uma assimilação de caminhos (PR, do inglês *path-relinking*) , várias soluções potenciais podem ser esquecidas durante o processo. Além disso, por causa da variabilidade aleatória das soluções que não é bem controlada, algumas vizinhanças podem voltar a ser investigadas mesmo que não ofereçam um bom potencial e outras podem nem mesmo

passar pelo processo de busca.

A fim de tentar contornar essas dificuldades na heurística ALNS sugere-se a hibridização com uma heurística cuja a implementação pode agregar grandes benefícios. A escolha da heurística CS nessa abordagem é devido ao mecanismo de agrupamento iterativo que pode ser visto como um processo de aprendizagem em torno de um ambiente externo (espaço de busca), de onde o conhecimento amostrado é adquirido, resultando em mudanças no aprendizado anterior. Outra vantagem do CS é a pertubação que gera um processo de assimilação importante e pode ser feito pelo método PR. Em suma, a heurística CS traz ao ALNS um processo local de busca inteligente e controlado.

Uma visão geral da heurística híbrida ALNS-CS pode ser definida pelo fluxograma apresentado na Figura 4.9. Nele encontramos a lógica completa da movimentação de soluções entre módulos assim como a ordem de utilização de cada componente. Nas subseções a seguir podem ser encontrados os detalhamentos de cada módulo. Em resumo inicia-se com a técnica da utilização de múltiplas soluções iniciais e escolhemos a primeira s_i; em seguida a heurística geradora ALNS é ativada gerando uma solução s_k e também sendo responsável pela criação de centros de grupos de soluções c_i; passamos ao processo de assimilação por PR seguido do agrupamento de soluções; um módulo de análise verifica se existe um grupo promissor e por fim a busca local com ALNS modificado é realizada e, em caso de ineficiência, uma nova solução inicial é inserida.

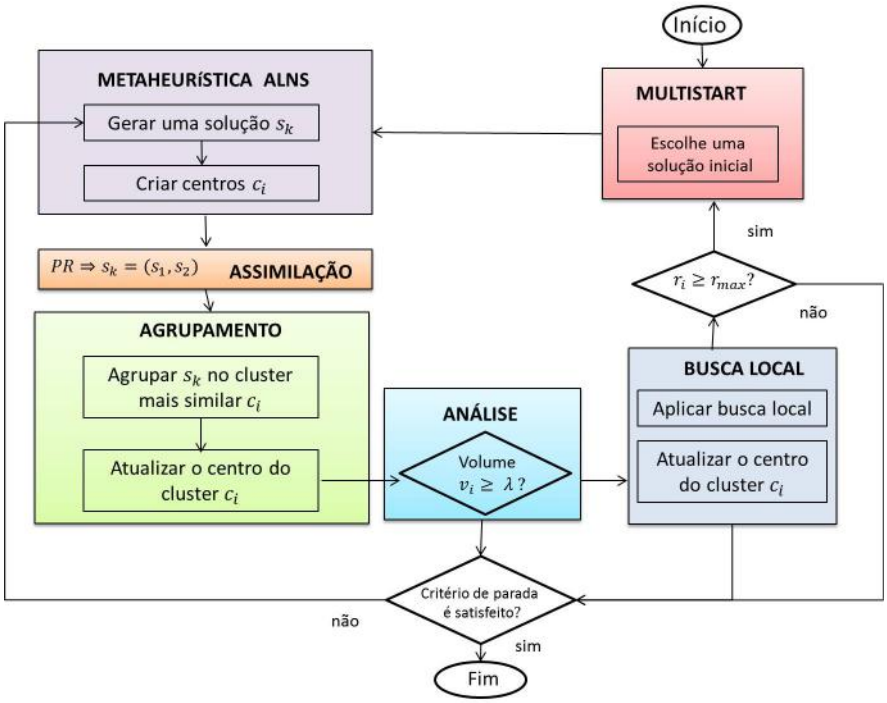


Figura 4.9: Fluxograma geral da heurística híbrida ALNS-CS

4.4.3 Métrica de distância

Para que a heurística híbrida possa agrupar soluções e verificar a diversidade entre elas é necessário definir alguma forma de medir a distância entre duas soluções. Sendo assim, uma função de medida de distância $d(i, j)$ é definida, a priori, para calcular a distância entre duas soluções como um número positivo, o qual é maior ou menor dependendo de quão distantes estão as duas soluções. Várias métricas de distância podem ser utilizadas, tais como, distância Euclidiana, distância de Manhattan, distância de Hamming, distância de Levenshtein, entre outras.

Neste trabalho utiliza-se a distância Hamming [66], uma vez que esta métrica possui um bom custo benefício em termos de qualidade e tempo computacional para calcular a distância entre duas soluções de um problema de otimização combinatória. A distância Hamming entre duas soluções s_1 e s_2 é definida como número de posições nas quais s_1 e s_2 são diferentes. A Figura 4.10 apresenta um exemplo do cálculo da distância de Hamming entre duas soluções com representação de uma rota feita por uma embarcação e 5 clientes. A distância precisa ser definida para cada problema abordado levando-se em consideração a representação adotada e suas características. Nesse caso são consideradas diferenças entre arcos ligando por exemplo UM_n e UM_m de uma solução e outra.

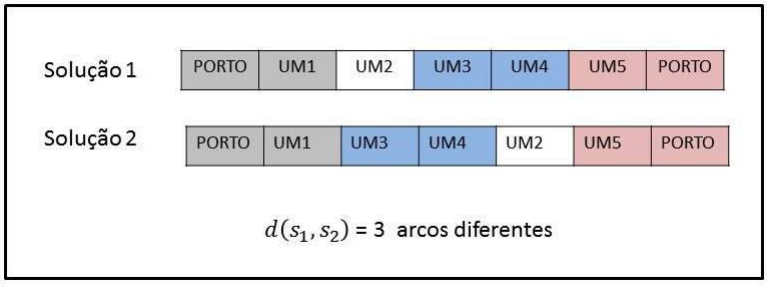


Figura 4.10: Exemplo de cálculo de distância Hamming

4.4.4 Múltiplas soluções iniciais

A fim de propor pontos de pertubação controlada ao longo das iterações da heurística híbrida traz-se uma mudança na construção de soluções iniciais. Diferentemente da implementação feita para heurística ALNS, a heurística híbrida propõe a utilização de múltiplas soluções iniciais que são inseridas em módulos específicos de acordo com o andamento do processo de busca. Para tanto, faz-se o uso do mesmo algoritmo de varredura descrito na Subseção 4.3.3, mas com algumas diferenças. Ao invés de gerar somente uma solução ao escolher um ponto fixo (no caso o porto) e criar rotas adicionando clientes de forma a seguir uma ordem crescente ou decrescente de seus ângulos, cria-se rotas escolhendo o primeiro cliente da rota de forma aleatória

e então seguimos no sentido horário ou anti-horário até que a inserção de clientes a esta rota seja limitada por alguma restrição imposta ao problema. A seguir o algoritmo volta a escolher um novo cliente de forma aleatória e qual sentido seguir agrupando clientes não roteados. Para mais detalhes vide Algoritmo 13.

Algorithm 13 Algoritmo de varredura aleatório

1: Inicializa: $K \leftarrow$ capacidade total, $T_t \leftarrow$ Tempo total;
2: Atribua um vértice base inicial $p \leftarrow 0$;
3: **enquanto** Existir um vértice i não agrupado **faça**
4: Escolha um veículo j não utilizado;
5: Escolha um cliente aleatoriamente e trace uma semi-reta com o vértice p;
6: Escolha seguir com a semi-reta no sentido horário ou anti-horário;
7: **enquanto** $\sum_i demanda_i < K_j$ e $\sum_i tempo_i < T_t$ **faça**
8: Movimente a semi-reta na direção escolhida acrescentando um vértice i a rota j;
9: **fim enquanto**
10: **fim enquanto**
11: **retorna** solução inicial s

Visto que é importante o controle do espaço de busca, essa inserção de soluções geradas pelo algoritmo de varredura é realizada de forma controlada. As soluções iniciais servem para gerar uma pertubação levando em consideração que tem-se como objetivo atingir espaços de busca diferentes. Para isso é elaborada uma lista de múltiplas soluções iniciais escolhidas a partir do conceito de diversidade máxima, mas que para este caso são consideradas soluções com uma porcentagem mínima de diferença quando calculada a distância entre elas. Esse algoritmo chamado de garantia da diversidade mínima tem como base o método construtivo proposto por SILVA *et al.* [67] e é descrito a seguir.

Seja N um conjunto de $n = 100$ soluções aleatórias, $d(i, j)$ a distância entre as soluções i, j de N, e M um conjunto diverso com m soluções ($m < n$). O método para gerar a lista com múltiplas soluções iniciais consiste em selecionar m soluções de N, de forma a maximizar a soma das distâncias entre essas soluções. Neste caso a lista é gerada pelo algoritmo de varredura aleatória descrito anteriormente 13. Inicialmente é necessário montar uma matriz simétrica $D(n \times n)$ que armazena as distâncias entre cada uma das soluções de N, preenchida da seguinte maneira: $d(i, j) = d(j, i)$ e $d(i, i) = 0, \forall i, j \in N$. O valor da diversidade de M é determinado pela soma das distâncias entre as soluções selecionadas para esse conjunto.

O método construtivo baseia-se na heurística de inserção mais distante. A primeira solução do conjunto $M(m_1)$ é escolhida aleatoriamente entre todas as soluções pertencentes a N. A segunda solução a ser selecionada deve ser a solução

$j \in N - (m_1)$ que forneça uma diversidade entre m_1 e j maior que ϵ que representa nessa trabalho 60% da quantidade de arestas contidas na solução. A partir da terceira solução é necessário calcular $D_{soma}(i), \forall i \in N - M$, que é a soma das diversidades entre o candidato i e todas as soluções pertencentes ao conjunto parcial M. A solução i que tiver o valor de D_{soma} com uma diversidade maior que ϵ é selecionada para ser incluída no conjunto M. O processo é repetido até que m soluções sejam selecionadas. Cada solução do conjunto M será uma solução inicial nas múltiplas inserções. O código do método é apresentado no Algoritmo 14 adaptado de SILVA *et al.* [67].

Algorithm 14 Processo de garantia de diversidade mínima

```
Gere um conjunto de n soluções aleatórias (N)
M = ∅
Escolha aleatoriamente a primeira solução m_1 ∈ N, M ← M ∪ m_1
para j ∈ N − m_1 faça
    calcule d(j, m_1)
fim para
m_2 = d(k, m_1), k ∈ N − m_1
se d(k, m_1) ≥ ε então
    M ← M ∪ {m_2}
fim se
para c = 3 até m faça
    Calcule D_soma(i), ∀i ∈ N − M
    m_c = D_soma(k), k ∈ N − M
    se D_soma(k) ≥ ε, k ∈ N − 1 então
        M ← M ∪ {m_c}
    fim se
fim para
retorna M
```

4.4.5 Criação de grupo de soluções

Primeiramente, é necessário definir o número de grupos de soluções utilizado na detecção de áreas promissoras, ou seja, a quantidade de regiões na qual o espaço de busca será dividido. Uma maneira de determinar o número de grupos é empiricamente, o que pode ser considerado arriscado se não for bem implementado. Se a escolha for um número pequeno de grupos a procura de soluções pode ser feita em poucas regiões do espaço de busca, enquanto que um número muito grande pode tornar o processo de agrupamento muito caro computacionalmente. A quantidade máxima de centros é definida por $n_c = 10$ através da análise feita a seguir.

Os grupos de soluções iniciais G_i são criados contendo uma solução chamada centro c_i que representa sua localização no espaço de busca. Inicialmente os valores

de volume v_i e o índice de ineficácia são os mesmos para todos os grupos, sendo $v_i \leftarrow 1$ e $r_i \leftarrow 0$.

A escolha dos centros deve ser feita de forma a representar diferentes regiões do espaço de busca. Em função disso, é proposto um método de criação dos centros dos grupos iniciais de forma que haja uma porcentagem mínima de diversidade entre eles em equilíbrio com a quantidade de soluções geradas considerando um dado número de iterações. A medida de diversidade entre dois grupos pode ser representada pela distância entre os dois centros desses grupos.

O problema de diversidade máxima é classificado como NP-hard [68] e sua resolução de forma ótima pode conduzir a um tempo computacional muito alto. Entretanto o deslocamento dos centros de grupos no espaço de busca ao longo das iterações torna suficiente a descoberta de soluções inicias consideradas boas e que abrangem diferentes áreas de acordo com a porcentagem de diversificação escolhida.

Sendo assim, a escolha para este trabalho foi a construção de grupos no decorrer da heurística híbrida. As soluções representando os centros são geradas pelo próprio algoritmo de otimização independente do ALNS gerador descrito pelo Algoritmo 16 e que tem como avaliação a métrica de distância Hamming com a garantia de uma porcentagem mínima de diversidade obtida pelo mesmo processo descrito anteriormente no Algoritmo 14. Uma vantagem disso é que no decorrer do processo esses grupos seguem a construção de soluções da heurística ALNS, que já possui diferentes operadores gerando centros em diferentes vizinhanças com uma diferença mínima entre centros.

4.4.6 Metaheurística ALNS geradora e processo de agrupamento

Uma metaheurística precisa trabalhar como um gerador de soluções para o processo de agrupamento. É importante que esta seja capaz de gerar um grande número de soluções diferentes, possibilitando uma análise ampla do espaço de busca. Em outras palavras, a meta-heurística precisa levar em consideração principalmente o critério de diversificação. O algoritmo ALNS gerador (Algoritmo 16) se encaixa perfeitamente nos requisitos para geração de boas e variadas soluções no espaço de busca. Além disso ele é executado independentemente dos demais componentes, sendo assim, as soluções armazenadas nos centros dos grupos não causam interferência no processo de busca da metaheurística, apenas agregam valor.

O ALNS gerador possui as mesmas características do algoritmo ALNS original descrito no Algoritmo 11, mas inclui a geração de centros c_i de grupos de soluções e o módulo de assimilação de caminhos. Nesta nova etapa, se a solução em avaliação s' for melhor que a atual s, verifica-se se o número de grupos formados até o momento

está abaixo da quantidade estabelecida n e se a diversidade mínima ϵ é garantida então atribui-se mais um centro $c_{i+1} = s'$.

O processo de agrupamento trabalha como um módulo classificador, conservando no sistema somente informações relevantes e direcionando a busca para regiões supostamente promissoras. O objetivo é reunir soluções similares dentro de grupos, mantendo uma solução como centro do grupo que seja representativa para as demais soluções. Para evitar um esforço computacional extra, o agrupamento é desenvolvido com um processo iterativo, no qual os grupos são progressivamente alimentados por soluções geradas em cada iteração da metaheurística.

A similaridade entre uma solução e os grupos é calculada pela medida de distância inicialmente definida. As soluções geradas pela metaheurística são enviadas para o processo de agrupamento a cada segmento de iterações ϕ e procura agrupá-las no grupo mais similar, ou seja, grupo que possuir a menor distância entre o centro c_i e a solução gerada pela metaheurística s_k é considerado o grupo G_i mais similar. Caso dois ou mais grupos tenham a mesma distância para a solução s_k, o grupo é escolhido aleatoriamente. É importante lembrar que as soluções s_k não são armazenadas nos grupos, apenas o volume do grupo é aumentado em uma unidade ($v_j = v_j + 1$).

O processo de assimilação consiste em atualizar os centros com a nova solução e procurar outras soluções que estejam ao redor. Segundo OLIVEIRA e LORENA [9], pode-se utilizar três formas diferentes de assimilação:

- Assimilação simples: insere uma porcentagem, definida a priori, das características da solução s_k no centro c_i;
- Assimilação por recombinação: realiza um cruzamento entre o centro c_i e a solução s_k gerando um novo centro com características de ambas as soluções e
- Assimilação por caminho: analisa um conjunto de soluções que são geradas no caminho que interconecta o centro c_i e a soluçãos_k.

Uma única solução é gerada nas assimilações simples e por recombinação, em contrapartida na assimilação por caminhos podem ser geradas várias soluções e o centro do grupo será a melhor solução encontrada. O fato de deslocar o centro para a melhor solução faz com que esta seja um bom ponto de partida para a busca local, necessitando de um número pequeno de modificações para encontrar um ótimo local ou global.

O método de Reconexão de caminho (PR) é utilizado para esse processo de assimilação e é mostrado em GLOVER e LAGUNA [69]. Ao realizar movimentos exploratórios na trajetória que interconecta duas soluções intensifica e diversifica a busca dentro do grupo. Esse método começa a partir do centro do grupo mais próximo c_i e da solução s_k que vem do componente MB. O procedimento começa

calculando a diferença simétrica entre as duas soluções, ou seja, o número de movimentos necessários para alcançar s_k de c_i. O movimento m^* resulta na melhor solução escolhida para ser executada em s_k. O conjunto de movimentos disponíveis é atualizado. O processo termina quando não há mais movimentos a serem explorados e a melhor solução encontrada no caminho é retornada pelo algoritmo como um novo centro ou quando uma porcentagem do caminho é analisada. Consulte Algoritmo 15 para obter mais detalhes.

Algorithm 15 Path-relinking

Inicializa: c_i e s_k
$s_{inicial} \leftarrow c_i$
$s_{guide} \leftarrow s_k$
Calcule as diferenças simétricas $\Delta(s_{inicial}, s_{guide})$
$s = s_{inicial}$
$f^* = \infty$
enquanto $\Delta(s, s_{guide}) \neq 0$ **faça**
    Analise todos os movimentos m
    Encontre o melhor movimento m^* e aplique em s
    $s \leftarrow s'$
    Atualize as diferenças simétricas $\Delta(s', s_{guide})$
    se $f(s') < f^*$ **então**
        $f^* = f(s')$
        $s^* = s'$
    fim se
fim enquanto
retorna s^*

Devido a facilidade e o baixo tempo computacional para aplicação desse método na heurística híbrida, são feitas em três inserções em fases e módulos diferentes. A primeira assimilação é feita quando a heurística geradora ALNS (Algoritmo 16) encontra uma melhor solução, essa trajetória conecta a melhor solução encontrada s_{best} ao grupo mais distante. Essa implementação visa ligar soluções o mais diferente possível, procurando rapidamente em espaços de busca não pesquisados. A cada término da geração de soluções s_k, o PR também é utilizado a fim de fazer uma conexão entre a solução gerada s_k e o centro mais próximo, como já é utilizado no CS. E por fim, dentro do módulo de BL (Algoritmo 17), quando a heurística encontra uma melhor solução s_{best}, o PR é usado para conectar a nova solução com a melhor solução anterior a ela. Essa última aplicação visa acelerar a busca local, ao procurar soluções no mesmo espaço de busca.

Após realizar a assimilação da solução s_k gerada pelo módulo MB é preciso conduzir a análise do grupo ativado, verificando se este pode ser considerado promissor. Esse processo é realizado pelo módulo BL nesse grupo.

Algorithm 16 Metaheurística ALNS geradora

```
1: Inicializa: s solução atual;
2: enquanto iter < φ faça
3:    s' ← s;
      Aplicar os operadores de destruição e reparação à s' e atualizar o número de
      vezes que eles foram usados.
4:    se z(s') < z(s) então
5:       s ← s';
6:       se quantidade de grupos i < n então
7:          aplica processo de diversidade mínima;
8:          se diversidade ≥ ϵ então
9:             Cria grupo G_{i+1} com centro c_{i+1};
               c_{i+1} ← s';
10:         fim se
11:      fim se
12:      se z(s) < z(s_best) então
13:         s_best ← s;
            atualize a pontuação para os operadores usados com π_1;
            PR(s_best, −c_i);
14:      senão
15:         atualize a pontuação dos operadores usados com π_2;
16:      fim se
17:   senão
18:      se s' é aceita pelo critério do SA então
19:         s ← s';
            atualize a pontuação para os operadores usados com π_3;
20:      fim se
21:   fim se
22:   iter ← iter + 1;
23: fim enquanto
24: retorna s_k;
```

4.4.7 Busca local

Em problemas de otimização as heurísticas de busca local (ou heurística de refinamento) constituem uma família de técnicas baseadas na noção de vizinhança. Essas heurísticas partem de uma solução inicial e caminham, a cada iteração, de vizinho para vizinho de acordo com a definição de vizinhança adotada.

Essa definição de vizinhança é importante em uma heurística de busca local. A partir de uma solução s do espaço de busca, deve ser sempre possível atingir qualquer outra solução em número definido de passos, utilizando um determinado tipo ou tipos de movimento.

Visto isso, optou-se por utilizar a própria heurística ALNS, que já mostra um bom desempenho e diversificação na estrutura de vizinhanças, mas com algumas modificações se comparada à aplicação anterior no MB gerador. Uma alteração importante é não utilizar operadores aleatórios, já que visamos uma busca em profundidade na região considerada promissora. E outra tão importante quanto e com o mesmo propósito, é não utilizar o critério de aceitação da solução s com uma certa probabilidade, ou seja, só são aceitas melhores soluções.

Além disso, a fim de obter uma reconfiguração maior na re-inserção de clientes na solução, considera-se uma mudança quanto à quantidade de clientes removidos. Como foi explicado anteriormente a remoção de uma grande quantidade de clientes pode gerar um alto número de combinações possíveis para os operadores de inserção, entretanto essa remoção é feita somente quando o módulo de BL é ativado, o tempo computacional não é fortemente afetado e aumentam as chances de se encontrar boas soluções. A última mudança é a inclusão do processo de assimilação entre a nova melhor solução encontrada s_{best} e a anterior a ela s_{best}^{-1} como explicado anteriormente na Seção 4.4.6.

O componente de busca local é um módulo de busca que visa a exploração de uma suposta região promissora, delimitada pelo grupo de soluções. Este processo acontece após ser descoberto um grupo de soluções promissor e uma intensificação da busca é aplicada à solução que representa o centro do grupo. Para se tornar promissor, um grupo precisa que seu volume v_i atinja um certo limitante λ, significando que provavelmente ele se encontra em uma boa região de busca. O índice de ineficácia r_i tem como função servir como memória caso a busca local não tenha obtido sucesso no centro c_i recentemente e é atualizado após a busca local ser executada. Para que a busca local tenha sucesso em um grupo é necessário que a solução encontrada seja a melhor solução s_{best} obtida até o momento dentro desse grupo.

A busca é intensificada no centro de um grupo considerado promissor, ou seja, $v_i \geq \lambda$ e $r_i < r_{max}$, aplicando a heurística de busca local ALNS modificada. Já o índice de inércia é zerado ou aumentado em uma unidade dependendo se a heurística encontra ou não uma nova melhor solução neste grupo.

Por outro lado, se o volume de um grupos de soluções atingir λ ($v_i \geq \lambda$) e o índice de ineficácia for alto ($r_i \geq r_{max}$), indicado que esta é uma região ruim ou que já foi suficientemente explorada pela heurística, aplica-se a uma pertubação no centro c_i. Essa pertubação é feita pelo algoritmo de varredura aleatório que escolhe múltiplas soluções iniciais a priori com uma distância suficientemente grande entre elas e visa permitir que o algoritmo escape dessa região e pesquise outras áreas do espaço de busca. Essa pertubação também se faz necessária em razão do CS utilizar um número máximo de grupos e do MB gerador, que é o ALNS, só aceitar soluções aleatórias quando o operador aleatório é escolhido ou não aceitar mais soluções aleatórias quando sua temperatura está baixa. Sempre que o volume de um grupo atingir λ é preciso reduzir o volume ($v_i = 0$) para impedir que o mesmo grupo seja considerado novamente promissor já na próxima iteração em que uma solução for agrupada a ele.

A eliminação de centros é vista como necessária devido a tendência da heurística ALNS à permanecer gerando soluções na mesma vizinhança. Essa técnica ajuda o algoritmo a gerar novas soluções a partir do momento que o centro já foi suficien-

temente investigado. Neste trabalho foi definido que o componente de busca local pode ficar no máximo $r_{max} = 2$ ativações consecutivas do módulo BL sem obter sucesso em um centro. Para evitar que informações sobre o espaço de busca sejam perdidas por meio da eliminação de um grupo, optou-se por permitir a criação de centros de forma dinâmica durante toda a metaheuristica. Grupos considerados inativos ($v_i = 0$) também são descartados durante a primeira eliminação. O Algoritmo 17 mostra o código da implementação completa do módulo de busca local ALNS modificado.

Algorithm 17 Busca Local com ALNS modificado

```
1:  Inicializa:S espaço de busca;
    Grupo G_i com centro c_i é considerado promissor;
    G_j = S − G_i ;
    Lista L de soluções inicias s_i;
2:  enquanto iter < iter_max faça
3:    se iter < iter_max/2 então
4:      Quantidade de clientes removidos = n ∗ 2;
5:    senão
6:      Quantidade de clientes removidos = n;
7:    fim se
8:    s' ← s;
      Aplicar os operadores de destruição e reparação à s' com exceção dos aleatórios
      e atualizar o número de vezes que eles foram usados.
9:    se z(s') < z(s) então
10:     s ← s';
11:     i_noimprove ← 0
12:     se z(s) < z(s_best) então
13:       s_best ← s;
          atualize a pontuação para os operadores usados com π_1;
          PR(s_best, s_best^{-1});
          c_i ← s_best;
14:     senão
15:       c_i ← s';
          atualize a pontuação dos operadores usados com π_2;
16:     fim se
17:   fim se
18:   iter ← iter + 1;
19: fim enquanto
20: se |L| ≠ ∅ e s_best = s_best^{-1} então
21:   s' ← s_i;
22: fim se
23: se r_i ≥ r_max então
24:   elimina c_i;
25:   para G_j ∈ S faça
26:     se v_j = 0 então
27:       elimina G_j;
28:     fim se
29:   fim para
30: fim se
31: v_i ← 0;
32: s_k ← s';
33: retorna s_k;
```

4.4.8 Código e fluxograma

De maneira geral a heurística híbrida aproveita muitos aspectos positivos da heurística ALNS e adiciona a técnica de detecção de áreas promissoras provindas do CS. O Algoritmo 18 descreve com detalhes o caminho percorrido por uma solução s_k e as ações realizadas por cada módulo dependendo da região de busca. A heurística inicia com parâmetro temperatura inicial (T_0) e as variáveis de melhor solução (s_{best}) e solução atual (s_k), por conseguinte é construída uma lista de soluções inicias L geradas pelo algoritmo de varredura aleatório de forma a considerar um fator mínimo de diversidade entre elas. Uma solução inicial $s_i \in L$ é escolhida e em seguida o módulo de busca com o ALNS gerador é ativado. Esse módulo gera uma solução s_k que prossegue para o módulo de assimilação fazendo um PR entre s_k e o centro mais próximo. A busca local é ativada se um grupo G_j é considerado promissor, ou seja, volume $v_j \geq \lambda$. Por fim os pesos são atualizados a cada φ iterações assim como a temperatura T.

Algorithm 18 Heurística híbrida: ALNS-CS

```
1: Inicialize: Todos os pesos são unitários e pontuações zeradas.
2: Algoritmo de varredura gera soluções iniciais s_i ∈ L;
   s_best ← s ← s_i, T ← T_0.
3: enquanto τ > 0.01 faça
4:    Gera solução s_k pela metaheuristica ALNS geradora;
5:    Encontre o o grupo com centro c_i mais similar a s_k;
6:    Insere s_k dentro do grupo G_i (v_i ← v_i + 1)
7:    PR(s_k, c_i);
      Atualiza o centro c_i;
8:    para j ∈ Gj faça
9:       se v_j ≥ λ então
10:         Busca local (ALNS modificado);
11:         v_i ← 0;
12:      fim se
13:   fim para
14:   a cada φ iterações atualiza pesos e pontuações.
15:   T ← α_T * T;
16:   i ← i + 1;
17: fim enquanto
18: retorna s_best;
```

Depois de detalhado o funcionamento geral de todo o sistema agora a heurística híbrida é ilustrada com um fluxograma mais completo como pode ser visto na Figura 4.11. Nele pode-se observar todos os módulos e a sequência de iterações feitas tanto na solução corrente como quando uma nova melhor solução é encontrada pelo algoritmo de uma maneira mais clara.

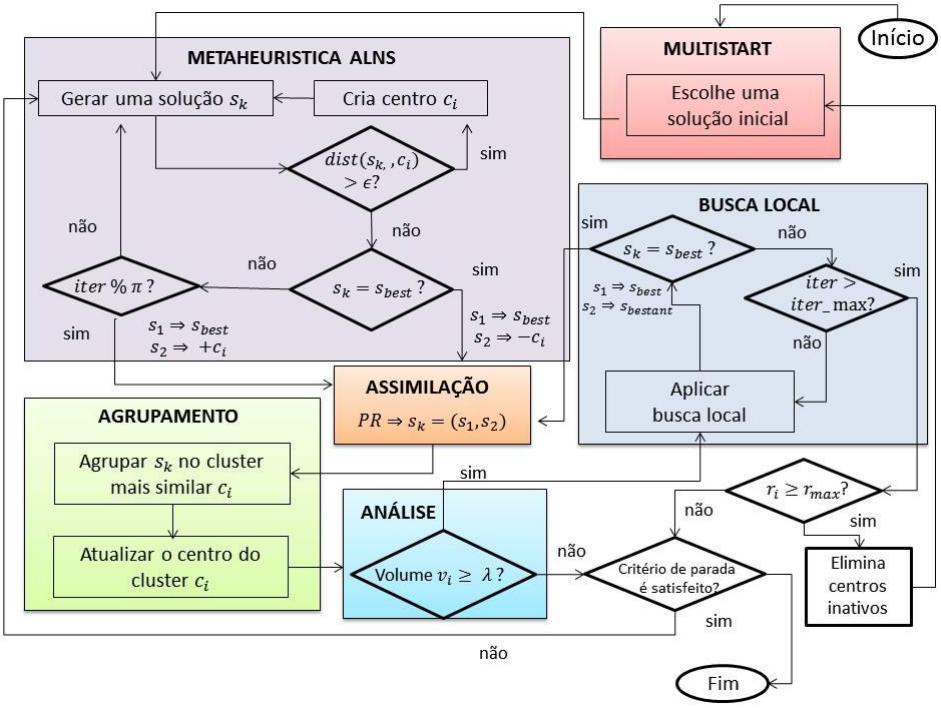


Figura 4.11: Fluxograma detalhado da heurística híbrida ALNS-CS

Capítulo 5

Experimentos computacionais

Neste capítulo os resultados computacionais da aplicação das heurísticas explicadas no capítulo anterior são apresentados. Além disso, descreve-se um conjunto de experimentos computacionais que foram realizados com o objetivo de estudar o comportamento das heurísticas diante das variações de seus parâmetros e especialmente com relação a cada componente do processo de agrupamento dentro da heurística híbrida formulada como tema principal.

Por meio dos resultados destes testes é possível determinar as principais características que as heurísticas precisam possuir para obter sucesso na resolução de problemas de otimização combinatória. Essas informações podem auxiliar futuras implementações das heurísticas para até mesmo outros problemas de otimização.

Todos os testes foram realizados com 10 execuções com quantidade de iterações variando de acordo com o tamanho da instância. Para efeito de ajustes, em alguns experimentos, considerou-se uma instância de cada tamanho a fim de obter uma boa calibração de parâmetros. O processo de calibração é executado de maneira linear como sugerido por PISINGER e ROPKE [7] para o ALNS.

a linguagem de programação C ++ é usada para resolver a heurística de agrupamento e a plataforma Aimms versão 3.13 para resolver métodos exatos. Para executar as heurísticas ALNS e híbrida foi usado somente a linguagem C++. O computador equipado com o processador Intel (M) Core i7 com 3.0 Ghz com até 8 Gb de RAM e o sistema operacional Windows 8.1.

Em resumo, o primeiro passo foi utilizar programação matemática para ter uma ideia de limitantes inferiores, melhores soluções encontradas guardando o tempo computacional e gap de cada instância. A metodologia seguinte trata-se de uma heurística de clusterização combinada a duas fases que utilizam método exato visando reduzir o tempo computacional. A fim de reduzir o número de etapas e também o tempo computacional da heurística clusterização-roteamento propõe-se a utilização da heurística ALNS que trata o problema em instâncias separadas e sugere um custo ainda menor ao tratar instâncias unificadas. Por fim a heurística

híbrida foi implementada e seus resultados são comparados aos resultados das outras implementações.

5.1 Instâncias

Os conjuntos de dados cedidos pela empresa são divididos em três classes: UR (UMs do Rio de Janeiro), UC (UMs de Campos) e Sondas. Esta divisão entre UR e UC é administrativa e é usada para agrupar antigas e novas plataformas de petróleo; portanto decidimos tratar o problema respeitando essa divisão e além disso propomos tratar todos os clientes ao mesmo tempo, a fim de avaliar qual impacto da divisão administrativa. Dessa maneira, criamos instâncias maiores (UT) , combinando dois conjuntos de dados UR + UC que possuem 2 atendimentos por semana. As sondas por possuírem 3 atendimentos por semana não são incluídas nesse estudo.

O tamanho das instâncias escolhidas para os testes varia ao longo do tempo seguindo vários fatores, como a necessidade de mais manutenção ou se UMs não estiverem operacionais por um período de tempo. Em geral, temos 3 instâncias da categoria UR com 18 e 19 UMs, 3 instâncias da categoria UC com de 40, 41 e 42 UMs e 3 instâncias combinadas com 59 e 60 UMs. Existem dois clientes com restrições de serviço durante o dia e um durante a noite. A capacidade máxima dos barcos é de 600 m ^{2}. A duração de um período (dia e noite) dura 12 horas.

5.2 Programação matemática

A metodologia ideal para critérios comparativos contundentes é a obtenção das soluções exatas para o problema. Entretanto para problemas que lidam com instâncias reais do PRESP, o tempo computacional sobe exponencialmente e o algoritmo exato não se mostra capaz de encontrar uma solução viável para a menor instância real com 18 clientes em até 24h de busca.

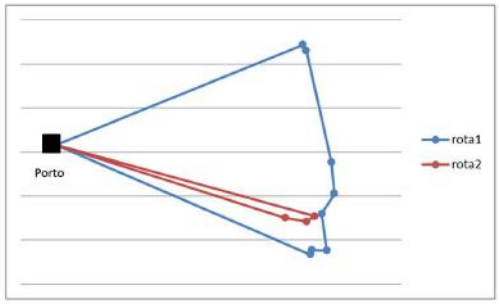
A fim de estudar o comportamento do método exato e verificar o impacto da quantidade de clientes, foram escolhidos 10 clientes da instância UR-1 como teste inicial e acrescentou-se clientes gradualmente até que o método não encontre uma solução viável em até 2,5 horas de tempo computacional.

Resultados da aplicação do método exato podem ser vista na Tabela 5.1. Nela encontram-se os resultados do tempo computacional obtido para o aumento gradual de clientes assim como o custo da sua melhor solução e gap.

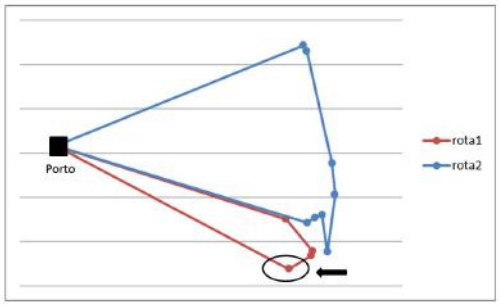
A Figura 5.1 ilustra a rota com 11 clientes (5.1 (a)), e sucessivamente quando se acrescentam um cliente por vez, gerando instâncias com 12 (5.1(b)), 13 (5.1 (c)) e 14 clientes em (5.1(d)) . Se observarmos na Tabela 5.1 a adição destes clientes tem um alto custo computacional.

	Aumento gradual de clientes		
Quantidade clientes	Melhor solução	Gap(%)	Tempo(s)
10	1538.31	0	50
11	1539.65	0	96
12	1543.08	0	6185
13	1641,6	1,43	9000
14	1686,62	3,03	9000

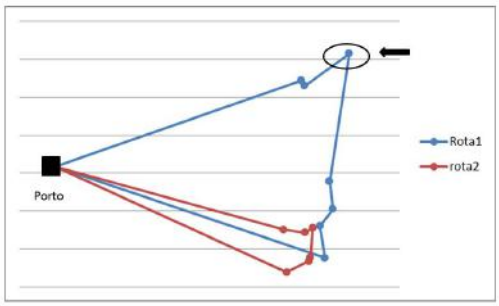
Tabela 5.1: Comparativo do aumento gradual de clientes para a instância UR-1



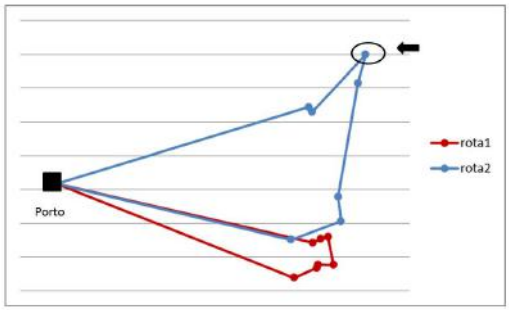
(a) Rota com 11 clientes



(b) Rota com 12 clientes



(c) Rota com 13 clientes



(d) Rota com 14 clientes

Figura 5.1: Diferença entre rotas com o acréscimo gradual de um cliente

Após realizada essa fase de teste podemos observar que a partir de em média 15 clientes o método exato não consegue mais encontrar soluções viáveis em até 2,5h. Uma análise das soluções obtidas sugere que o método tem dificuldades em gerar boas soluções viáveis, isto é bons limitantes superiores. O gráfico da Figura 5.2 mostra o exemplo do comportamento desses limitantes para uma instância com 14 clientes, escolhida por ser a última em que o método encontra soluções viáveis. Os resultados dos limitantes inferiores (LI) encontrados para cada instância são explicitados na Tabela 5.2 para cada instância em função do tempo computacional.

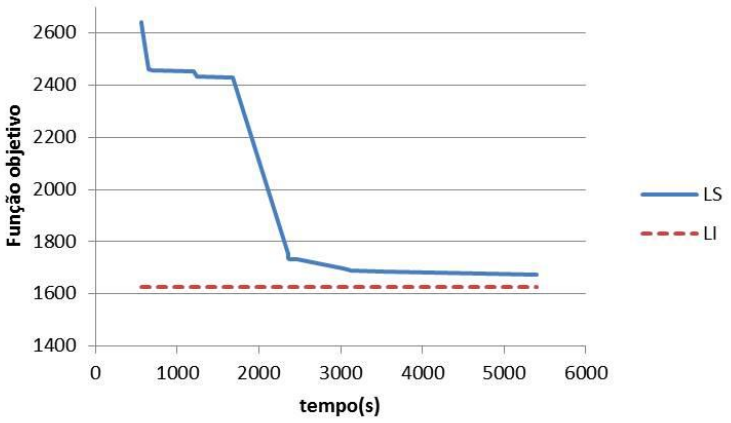


Figura 5.2: Visualização do comportamento dos limitantes superior (LS) e inferior (LI) para instância UR-1 com 14 clientes

	Limitantes inferiores
instância	LI
UR-1	1722.70
UR-2	2216.02
UR-3	2297.82
UC-1	3246.90
UC-2	2998.26
UC-3	3177.62
UT-1	4375.41
UT-2	4526.54
UT-3	5067.29

Tabela 5.2: Valores dos limitantes inferiores encontrados pelo método exato

5.3 Clusterização e roteamento

Visto que o método exato não encontra soluções viáveis em um tempo computacional de até 2,5h em uma instância com 15 clientes a primeira proposta para a resolução do problema é a implementação do método conhecido como clusterização-roteamento. Como foi visto anteriormente esse método realiza a divisão das instâncias em pequenos grupos de clientes por uma heurística de clusterização e a aplicação posterior do método exato nesses pequenos grupos de clientes gera boas soluções para o problema.

Infelizmente a comparação dos resultados não pode ser feita com os resultados obtidos pelo método exato, já que não encontra soluções viáveis. Alternativamente faz-se uma comparação com as soluções que eram usadas pela empresa na época a que se referem as instâncias.

5.3.1 Clusterização de clientes

A divisão em pequenos grupos de clientes é feita pela heurística de arrependimento como visto na Seção 4.2.1. Para que essa partição tenha um bom desempenho é preciso decidir os valores de alguns parâmetros que possuem papel importante no momento da aplicação do método exato para o roteamento. Esses parâmetros são: o número de grupos (NG) que provem da divisão da quantidade de clientes e o número máximo de iterações da heurística (IH).

A calibração feita para o parâmetro que controla a quantidade de iterações da heurística (IH) mostra que a partir de 200 iterações a função objetivo já apresenta uma boa estabilidade, contudo existe ainda uma pequena melhoria (diminuição) da mesma até em torno de 1000 iterações. O gráfico da Figura 5.3 ilustra o comportamento da pior e melhor soluções geradas pela heurística de arrependimento assim como a solução média depois de 10 replicações para um instância de porte médio (UC-1).

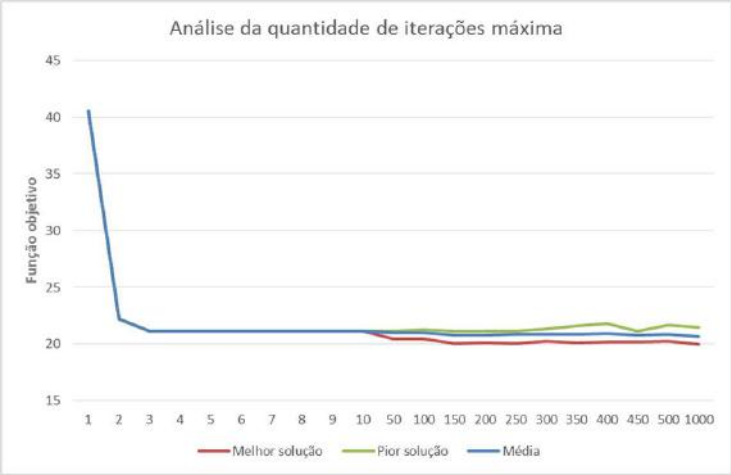


Figura 5.3: Calibração para quantidade de iterações da heurística de clusterização

As Figuras 5.4 e 5.5 ilustram para a instância UC-1 a evolução da função objetivo e o posicionamento espacial das UMs já agrupadas, reforçando a eficacidade da clusterização.

5.3.2 Roteamento e re-sequenciamento

Após a clusterização, cada instância está dividida em grupos menores e prontas para serem roteadas. Como explicado na Seção 4.2.2, nessa fase o problema completo não é resolvido, sendo consideradas somente restrições de tempo total de viagem e capacidade máxima da embarcação.

Um processo importante que precisa ser avaliado é o impacto da escolha da quantidade de iterações da heurística de clusterização nos resultados da primeira fase

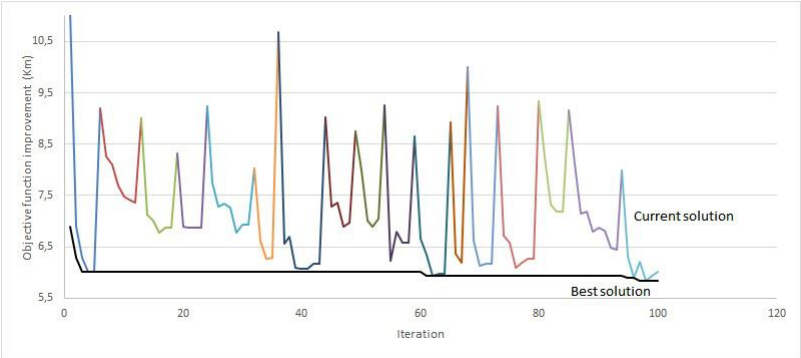


Figura 5.4: Evolução da função objetivo ao longo da iterações

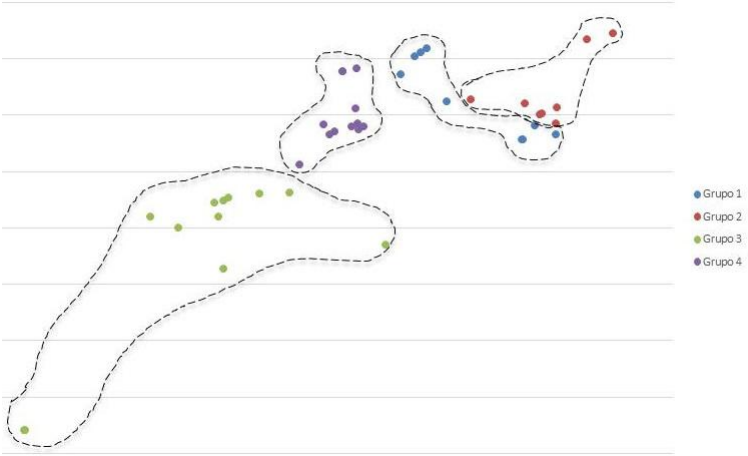


Figura 5.5: Visualização de clusters formados para instância UC-1

do roteamento, já que uma divisão de grupos mal realizada pode gerar soluções ruins. Os resultados de 5 replicações em um teste feito na instância UC-2 é representado no gráfico da Figura 5.6. Neste teste usamos diferentes número de iterações ($IH = 200$ e 1000) na heurística de clusterização e por conseguinte a média desses resultados encontrados na replicações com diferentes divisões de grupos (1 a 5 grupos) serve como entrada da primeira fase do roteamento. O coeficiente de variação para os valores encontrados fica abaixo de $coef < 0,001$ o que indica que com 200 iterações a heurística já gera bons resultados para o roteamento.

O número de grupos também precisa ser calibrado a fim de avaliarmos a relação entre a qualidade da solução encontrada e o tempo computacional utilizado na fase de roteamento para cada quantidade de grupos em que o conjunto de UMs é dividido. Os clientes são agrupados em no máximo 4 grupos. Os gráficos da Figura 5.7 mostram, por exemplo, resultados para três instâncias de diferentes tamanhos já a Tabela 5.3 apresenta os resultados do efeito da divisão em 1, 2, 3 e 4 grupos, para todas as instâncias. São apresentados o valor da solução, o gap e o tempo de execução para cada instância e cada número de grupos.

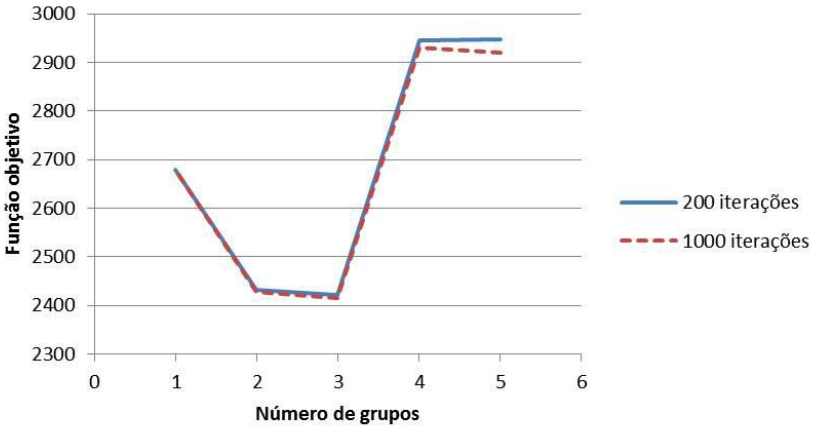
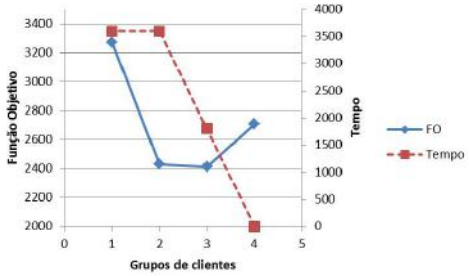
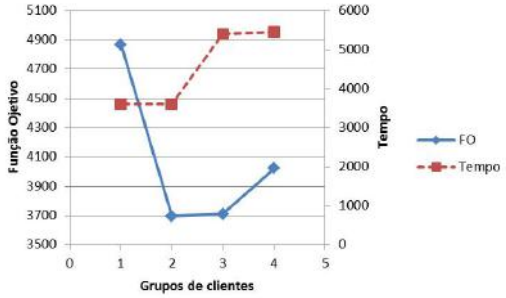


Figura 5.6: Comportamento médio das soluções encontradas pela heurística de arrependimento com $IH = \{200, 1000\}$



(a) Divisão em grupos UC-1

(b) Divisão em grupos UR-1



(c) Divisão em grupos UT-1

Figura 5.7: Dimensionamento de grupos

Podemos perceber diferentes comportamentos, mas em geral dividir os clientes em 2-3 grupos gera boas soluções em um baixo tempo computacional. Portanto, o número de grupos específico pra cada instância é escolhido de acordo com o tempo computacional mais baixo e a qualidade da solução.

	NG=1			NG=2			NG=3			NG=4		
instância	**sol**	**gap(%)**	**tempo(s)**	**sol**	**gap(%)**	**tempo(s)**	**sol**	**gap(%)**	**tempo(s)**	**sol**	**gap(%)**	**tempo(s)**
UR-1	1474.27	2.84	1800	1511.28	0	5.5	1962.39	0	1.5	-	-	-
UR-2	1607.02	24.25	1800	1623.35	0	25	1933.89	0	5	-	-	-
UR-3	1653.81	3.4	1800	1965.23	0	35	1964.18	0	5	-	-	-
UC-1	3269.70	40.59	3600	2431.44	5.7	3600	2413.44	1.4	1800	2708.19	0	5
UC-2	2790.48	43.31	3600	2602.40	19.96	3600	2641.19	30.05	1800	2918.30	39.8	1800
UC-3	2682.46	21	3600	2681.6	3.5	3600	2668.75	28	1800	3031.8	0	5
UT-1	4867.71	39.83	3600	3694.47	6.1	3600	3709.51	8.1	3600	4022.19	1.5	3600
UT-2	5176.92	51.46	3600	4231.48	35.7	3600	4262.11	23.3	3600	4459.78	24.9	3600
UT-3	18996.3	81.96	3600	4386.65	19.6	3600	4525.68	25.1	3600	4851.64	0	2400

Tabela 5.3: Resultado para o roteamento de acordo com o número de grupos após a clusterização

Após o roteamento de todas as instâncias, obtemos os resultados preliminares. Finalmente, o terceiro e último passo que é realizado após o roteamento é o re-sequenciamento. Nessa fase re-posicionamos as unidades de maneira que elas respeitem as restrições de tempo máximo entre viagens e horários de abertura tentando obter o menor tempo de espera possível.

A Tabela 5.4 mostra os valores da função objetivo (FO) do roteamento na coluna *"Rotear"* e do re-sequenciamento na coluna *"Re-sequenciar"* seguidos da coluna de tempo total e taxa de melhoria (*"Melhoria"*) se o resultado é comparado a solução da empresa. Nessa tabela também podemos ver que os resultados já se mostram satisfatórios e com custos menores se comparado as rotas realizadas pela empresa, além disto as rotas são obtidas de maneira automatizada, porém ainda com alto custo de tempo computacional.

Instância	n	Solução da empresa	**Rotear**	**Re-sequenciar**		
			FO	**FO**	**Tempo (s)**	**Melhoria (%)**
UR-1	18	2553.62	2538.54	2548.57	1800	2.89
UR-2	18	3266.31	3214.04	3215.66	1800	1.55
UR-3	19	4861.14	3307.62	3421.3	1800	29.61
UC-1	41	6048.9	4826.88	4828.75	3600	20.17
UC-2	42	6295.16	5204.80	5214.70	3600	24.26
UC-3	40	6185.72	5337.5	5344.86	3600	13.59
UT-1	59	8602.52	7388.94	7400.25	3600	13.97
UT-2	60	9561.47	8462.86	8471.77	3600	19.01
UT-3	59	11046.86	8773.3	8786.56	3600	20.46
Média		6569.07	5034.05	5056.55	3000	16.16

Tabela 5.4: Resumo de resultados, % melhoramento e tempo computacional para as instâncias UR, UC e UT

A Figura 5.8 mostra um exemplo de uma rota que ao ser sequenciada muda os clientes de posição buscando minimizar o tempo de espera e respeitar o tempo de ciclo.

Viagem 1								
Porto	1	2	3	4	5	6	7	Porto
Terça 20h	Quarta 7h	Quarta 19h	Quinta 8h	Quinta 12h	Quinta 18h	Sexta 1h	Sexta 8h	Sábado 14h
Viagem 2								
Porto	1	2	3	4	5	6	7	Porto
Sábado 6h	Sábado 17h	Segunda 5h	Segunda 18h	Segunda 23h +8h	Segunda 13h	Segunda 20h	Terça 3h	Terça 17h

Horário de chegada + tempo de espera para começar o turno

(a) Roteamento mantendo a mesma sequencia para as duas viagens

Viagem 1								
Porto	6	1	2	5	3	4	7	Porto
Terça 20h	Quarta 4h	Quarta 15h	Quinta 3h	Quinta 15h	Sexta 9h	Sexta 13h	Sexta 22h	Sábado 17h
Viagem 2								
Porto	6	1	2	5	7	3	4	Porto
Sábado 6h	Sábado 14h	Segunda 1h	Segunda 13h	Segunda 1h	Segunda 17h	Terça 8h	Terça 13h	Quarta 1h

Horário de chegada dentro do turno Tempo entre viagens dentro de 3,5 dias

(b) Re-sequenciamento respeitando as restrições de tempo de ciclo e minimizando os tempos de espera

Figura 5.8: Roteamento e re-sequenciamento

5.4 ALNS

Visando melhorar os resultados, e principalmente a redução do tempo computacional, foi desenvolvida a heurística ALNS aplicada ao PRESP. Através dessa implementação reduz-se os passos que eram realizados pela heurística de clusterização mais roteamento. Contudo, para uma boa confiabilidade do processo, a calibração de alguns parâmetros precisa ser realizada nessa heurística e essa etapa foi feita de maneira linear, ou seja, uma vez definido um parâmetro este não é alterado e os outros são ajustados. O processo de calibração foi aplicado a temperatura inicial (T_0), ao número de clientes a serem removidos (n), ao tamanho do segmento de iterações que atualiza os pesos (φ) e ao número máximo de iterações ($iter_{max}$).

5.4.1 Ajuste de parâmetros

- **Temperatura inicial:**

  Muito se discute sobre o impacto da escolha da temperatura inicial e como vimos na seção 4.3.2 a probabilidade de aceitação é maior quanto maior a temperatura. Logo se escolhemos baixas temperaturas a variabilidade do algoritmo será afetada. Testes foram feitos com 10 execuções para instâncias de

tamanhos diferentes UR-1,UC-1 e UT-1 fixando a temperatura inicial T_0 com grandezas equivalentes ao valor da função objetivo na solução inicial da menor instância até mais de 12 vezes a mesma visto que a instância de maior tamanho tem solução inicial 7 vezes maior. Foram analisados os valores: $5000, 15000, 30000, 60000$. O gráfico da Figura 5.9 mostra a média das melhores soluções de cada instância em função da variação da temperatura inicial. Podemos observar que as melhores soluções encontradas variando a temperatura inicial não sofrem uma mudança significativa independente do tamanho da instância, entretanto possui uma leve tendência a obter uma média de melhores soluções em $T_0 = 30000$ e por isso esse valor foi escolhido.

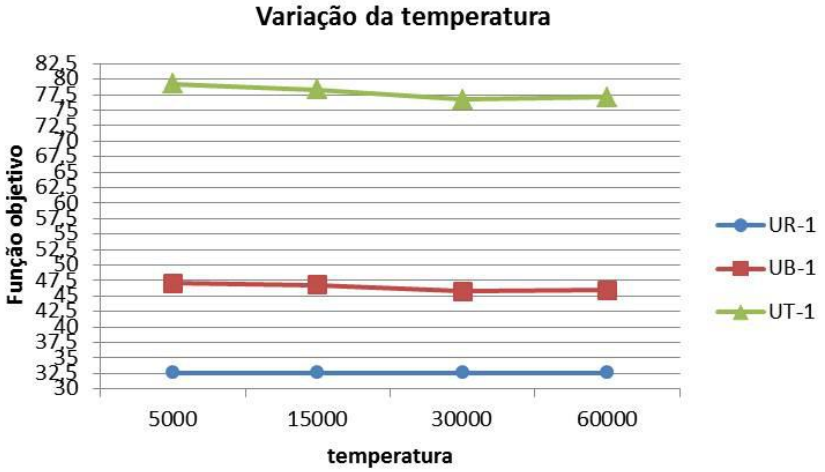
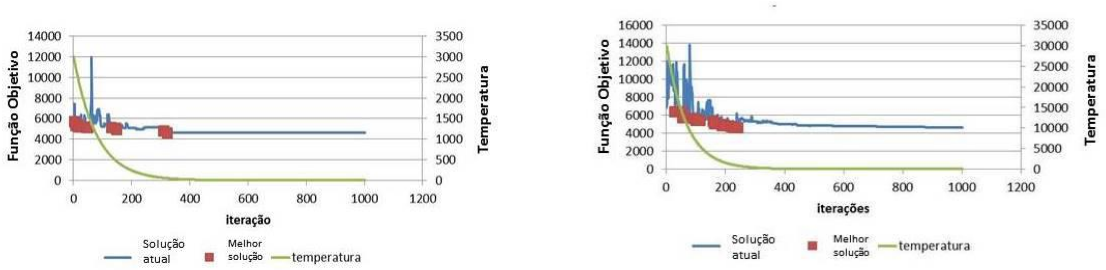


Figura 5.9: Análise do impacto da variação da temperatura inicial na heurística ALNS

Os gráficos da Figura 5.10 mostram o que acontece quando reduz-se muito a temperatura inicial, por exemplo um terço do valor da função objetivo para a instância UR-1 e quando considera-se uma temperatura inicial 5 vezes maior. Podemos concluir que quanto menor a temperatura inicial, menor a probabilidade de aceitação da solução pelo critério do *simulated anneling* explicado na seção 4.3.2 e maior as chances de não conseguir escapar de um ótimo local.



(a) Temperatura inicial $T_0 = 3000$ (b) Temperatura inicial $T_0 = 30000$

Figura 5.10: Análise da temperatura inicial

- **Impacto do percentual de destruição:**

  A análise deste parâmetro é importante pelo forte impacto que a porcentagem de destruição da solução (% destr) tem no tempo computacional devido a quantidade de re-inserções desses mesmos clientes na etapa de reparação como visto em ROPKE e PISINGER [37]. O parâmetro $\%destr$ é um número aleatório uniformemente distribuído sorteado dentro da faixa $[a\%, b\%]$. Foram avaliadas as seguintes faixas percentuais: para instâncias menores de até 21 clientes $[5\% - 25\%]$ e $[25\% - 50\%]$; para instâncias médias de até 42 clientes $[1\% - 15\%]$ e $[15\% - 25\%]$ e para instâncias maiores de até 61 clientes $[1\% - 8\%]$ e $[8\% - 15\%]$. Algumas percentagens não são nem consideradas devido ao alto tempo computacional.

  A Tabela 5.5 mostra o valor das melhores soluções geradas pelo ALNS e o tempo de processamento para diferentes percentuais de destruição para instâncias UR-2, UC-2 e UT-2. Esses resultados representam os valores médios de 10 execuções do algoritmo em $\{2000, 1000, 500\}$ iterações para UR, UC e UT respectivamente. Com esses dados fica claro que quanto maior o percentual de destruição melhores são as soluções encontradas, entretanto o tempo de processamento aumenta de forma rápida.

instância	% dest	FO	Tempo (s)
UR-2	[5% - 25%]	3201.31	548
	[25% - 50%]	3183.14	3401
UC-2	[1% - 15%]	5303.95	1557
	[15%-25%]	4921.25	9054
UT-2	[1% - 8%]	8480.86	1929
	[8%-15%]	8145.61	9847

Tabela 5.5: Comparação da função objetivo e tempo computacional com diferentes porcentagens de destruição

Portanto, com base na Tabela 5.5 o valor escolhido para as instâncias menores foi de $[5\% - 25\%]$, para instâncias médias $[1\% - 15\%]$ e para instâncias maiores $[1\% - 8\%]$.

- **Tamanho do segmento:**
  O tamanho dos segmento φ é um parâmetro importante pela sua influência sobre a atualização dos pesos dos operadores. Os valores escolhidos para avaliar os ajustes foram: $\varphi = \{100, 200, 300\}$.

  A Figura 5.11 mostra o valor da média das melhores soluções geradas pelo ALNS em função dos diferentes valores em φ para a instância UC-1 com variação na quantidade de iterações $1000, 2000$ e 3000.

  Pode ser observado que quando o tamanho do segmento está entre 100 e 200 obtemos melhores resultados na função objetivo. Isso acontece devido ao fato da atualização dos operadores acontecer mais vezes logo a probabilidade de um bom operador ser escolhido aumenta e por consequência gera melhores resultados.

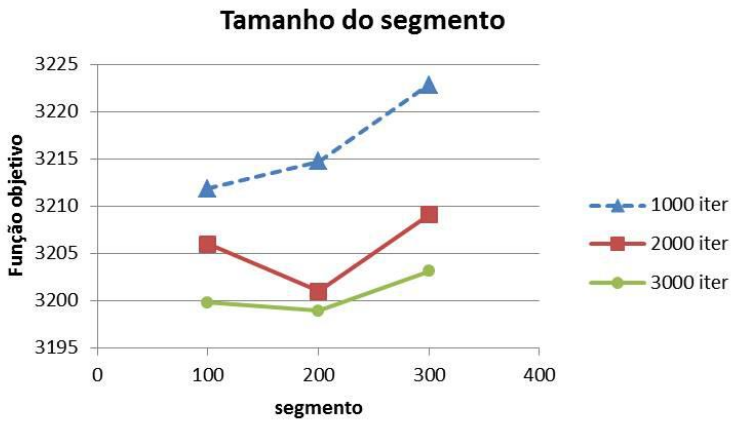


Figura 5.11: Análise da alteração do segmento em relação a função objetivo e quantidade de iterações

- **Número máximo de iterações:**
  A calibração do número máximo de iterações ($iter_{max}$) procura por um valor em que obtém-se uma boa solução em tempo computacional favorável. Os valores escolhidos foram diferentes para cada classe de tamanho de instância: $iter_{max} = \{1000, 2000\}$ para instâncias menores, $iter_{max} = \{1000, 1500\}$ para instâncias médias e $iter_{max} = \{500, 1000\}$ para instâncias maiores. Na Tabela 5.6 vemos como é de se esperar que o valor da função objetivo diminui ao aumentar o número de iterações, no entanto, como era esperado, o tempo computacional aumenta. Apesar disso, considera-se as melhorias importan-

tes e as maiores quantidades de iterações são escolhidas para cada grupo de instâncias UR, UC e UT ($iter_{max} = \{2000, 1500, 1000\}$).

instância	# de iterações	FO	Tempo (s)
UR-2	1000	3211.90	275
	2000	3206.60	317
UC-2	1000	5403.95	1557
	1500	5320.82	2008
UT-2	500	8480.86	1929
	1000	8308.09	3021

Tabela 5.6: Comparação da função objetivo e tempo computacional em diferentes número máximo de iterações

5.4.2 Resultados computacionais

Os resultados da aplicação da heurística ALNS mostraram-se ainda mais satisfatórios quando se espera boas soluções em um menor tempo computacional modelado em apenas uma heurística. A Tabela 5.7 mostra na primeira coluna os nomes das instâncias seguidas do número de UMs contidas. Ao lado são mostrados o custo da solução antes aplicada pela empresa, os novos custos obtidos pela heurística ALNS ("Best"o melhor valor encontrado e "Md"o valor médio em 10 execuções), o tempo de execução e o percentual de melhoria em relação a empresa.

Instância	n	Solução da empresa	**ALNS**			
			Best	**Md**	**Tempo (s)**	**Melhoria (%))**
$UR-1$	18	2553.62	2521.92	2538.87	319	1.24
$UR-2$	18	3266.31	3183.01	3206.60	317	2.55
$UR-3$	19	4861.14	3296.09	3307.23	277	32.19
$UC-1$	41	6048.90	4372.51	4595.46	1181	27.71
$UC-2$	42	6295.16	4937.06	5320.82	2008	21.57
$UC-3$	40	6185.72	5147.18	5457.18	1263	16.78
$UT-1$	59	8602.52	6896.67	7112.40	3590	21.11
$UT-2$	60	9561.47	8131.52	8308.09	3021	16.31
$UT-3$	59	11046.86	8331.08	8812.48	2259	24.58
Média		6569.07	5175.23	5409.54	1581	18.24

Tabela 5.7: Resumo de resultados, % melhoramento e tempo computacional para as instâncias UR,UC e UT

A Figura 5.12 representa a evolução do custo da solução em relação ao número de iterações, e observamos que, após cerca de 20 iterações, o algoritmo produziu uma solução melhor do que a da empresa. E esta solução ainda foi significativamente melhorada nas fases posteriores de nossa metaheurística.

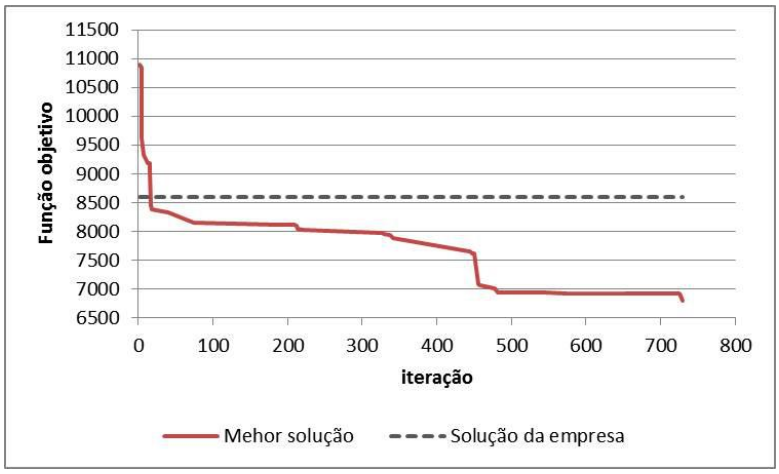


Figura 5.12: Melhores soluções: Empresa vs ALNS para instância $UT-2$

5.5 Heurística híbrida ALNS-CS

Nesta seção são mostrados os resultados gerais assim como os ajustes feitos nos parâmetros utilizados na heurística híbrida ALNS-CS. E exatamente por ser a hibridização de dois métodos, conta com mais parâmetros. Os resultados computacionais finalizam a seção mostrando o desempenho dessa abordagem.

5.5.1 Ajuste de parâmetros

Nosso algoritmo híbrido conta com os seguintes parâmetros que precisam de calibração: $iter_{max}$ que é o número máximo de iterações, a temperatura inicial T_0, número máximo de centros $|C|$, quantidade de soluções inicias $|M|$ e limitante de busca local λ.

Dado que o tamanho do segmento φ e a taxa de destruição das soluções que definem a quantidade n de clientes são parâmetros utilizados pelo ALNS e visto que já foram analisados na Seção 5.4.1 utilizaremos os mesmos valores.

- **Temperatura inicial** Assim como descrito na Seção 5.4.1 para o ALNS, também foram feitos testes para a calibração da temperatura inicial para a heurística híbrida. Os testes foram feitos com as mesmas três instâncias UR-1,UC-1 e UT-1 e foram usados os valores de temperatura inicial $T_0 = \{5000, 15000, 30000, 60000\}$.

  Mais uma vez, como podemos observar no gráfico apresentado na Figura 5.13, obtivemos também soluções com um coeficiente de variação baixo quando escolhemos temperaturas iniciais diferentes. Optamos por utilizar $T_0 = 30000$ devido a pequena queda na média de soluções.

- **Criação de soluções iniciais**

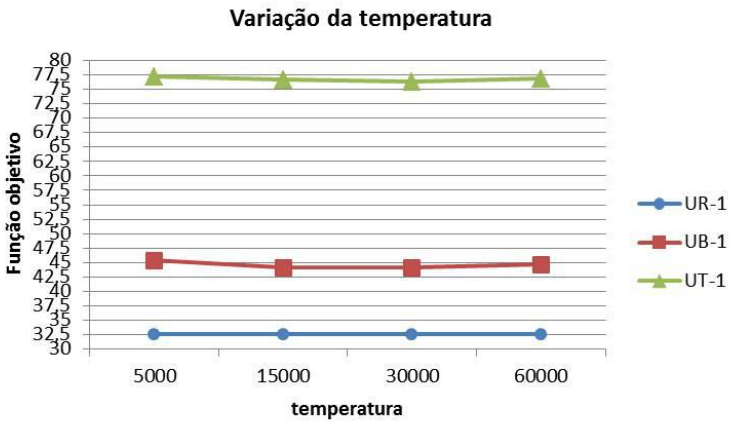


Figura 5.13: Análise do impacto da variação da temperatura inicial na heurística híbrida

As soluções são geradas pelo algoritmo de varredura aleatório e passam pela análise de diversidade. A quantidade de soluções inicias que são inseridas no conjunto M é escolhida em função da porcentagem de diferença entre soluções a fim de obter uma maior cobertura do espaço de busca ao aplicar essa pertubação ao algoritmo. Os testes foram feitos com porcentagens de diversidade acima de $50\%, 60\%, 65\%$ e 70%. Foi constatado que o número de soluções escolhidas $|M|$ dentro de um conjunto N de 1000 soluções para $50\%, 60\%$ 65% e 70% variam em média como os valores mostrados na Tabela 5.8. Portanto para que tenhamos soluções que cubram espaços de busca, sendo o mais diferente possíveis e ao mesmo tempo obtenha-se uma quantidade suficiente de soluções iniciais disponíveis depois de uma busca local mal sucedida, ou seja, após ativar o módulo de busca local e não encontrar melhores soluções (no mínimo 5 e no máximo 10 pertubações), escolhemos as porcentagens de diversidade entre soluções de 65%.

Grupo de instâncias	50%	60%	65%	70%
UR	30	10	8	4
UC	78	14	10	8
UT	100	18	12	8

Tabela 5.8: Análise do número de soluções iniciais

- **Desempenho com número diferentes de grupos de soluções**

  A quantidade de regiões na qual o algoritmo divide o espaço de busca é determinado pelo número máximo de grupos definido inicialmente. Desta forma, ter apenas um grupo significa tratar o espaço de busca como uma única região, e esse único grupo atingir o limitante λ a cada λ soluções geradas pelo pro-

cesso de agrupamento. Por outro lado, definir um número maior de grupos permite ao método descobrir quais regiões são promissoras intensificando as buscas somente nessas regiões.

Para analisar o comportamento da abordagem híbrida com diferentes números de grupos foi utilizada a instância (UC-1) e as quantidades foram testadas em função da diversidade de $40\%, 50\%$, 60% e 70% entre elas com um máximo de $|C| = 15$, já que são criadas ao decorrer do algoritmo.

Visto que uma maior diversidade gera menos possíveis centros e que uma porcentagem de diversidade muito baixa gera centros de grupos de soluções muito próximos no espaço de busca, procura-se escolher de forma a encontrar um equilíbrio e também que a busca possa se deslocar com facilidade entre os grupos.

A Figura 5.14 apresenta um gráfico com a eficiência do componente de busca local para as diferentes porcentagens de diversidade. Neste gráfico se observa que quanto maior é diversidade entre grupos, maior é a eficiência da busca local, ou seja a probabilidade de se escapar de um ótimo local é maior quando se divide o espaço de busca em mais regiões.

É possível também estabelecer uma análise sobre o número de centros já que quanto menos exigência sobre a diversidade, mais grupos são formados. Entretanto, se diminuímos muito a diferença exigida entre os centros, a busca local será feita em menor quantidade e com menores chances de sucesso já que se trata de regiões próximas que são investigadas repetidamente. Todavia se aumentamos muito a porcentagem de diversidade, o número de centros é muito baixo o que dificulta a a busca. Por isso a porcentagem escolhida foi de 60%.

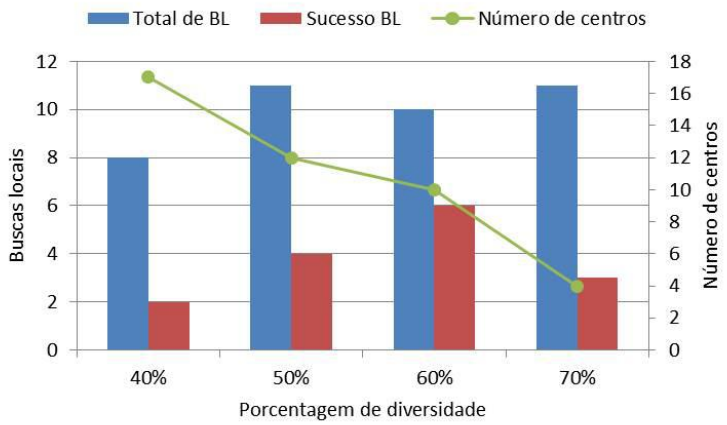


Figura 5.14: Influência da quantidade de centros na busca local

- **Desempenho com diferentes limitantes de busca local**

O valor do limitante λ tem influência na quantidade de vezes que o componente de busca local é executado. Portanto quanto maior o valor de λ, mais soluções serão necessárias para um grupo de soluções se tornar promissor e, então mais procedimentos de busca serão realizados.

Para analisar o desempenho da nossa heurística híbrida com diferentes valores de limitantes, foi usada a instância UR-3 com 10 execuções e com 1000 iterações. Os valores dos limitantes usados foram $\lambda = 5, 10, 15, 20$. Lembrando que quando λ é igual a 1 será aplicado busca no centro toda vez que a solução é gerada o que obviamente gera um tempo computacional muito acima do aceitável.

A Figura 5.15 ilustra o gráfico com o tempo computacional médio em função da variação do limitante λ. Como já era esperado, o tempo computacional do algoritmo para $\lambda = 5$ foi superior aos demais valores testados, o que é explicado pela maior quantidade de busca local realizada. Assim quanto maior o valor de λ menor é o tempo computacional da heurística híbrida. Nesse caso, estabilizando em $\lambda = \{15$ e $20\}$.

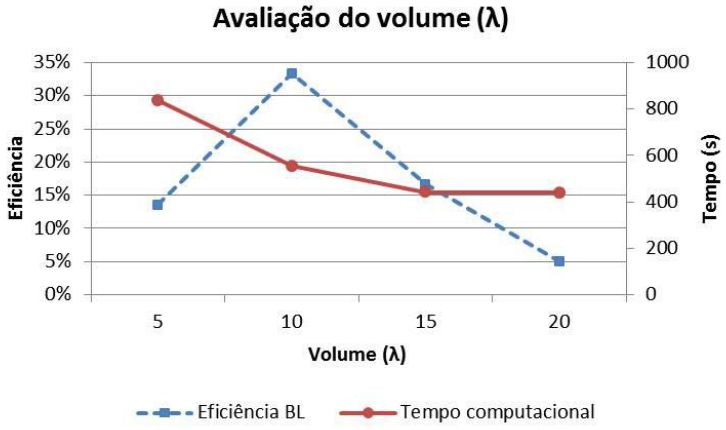


Figura 5.15: Gráfico com a análise da eficiência da busca local e tempo computacional variando a quantidade de grupos

Valores de $\lambda = \{15$ e $20\}$ possuem um baixo tempo computacional, entretanto, esses valores também produzem resultados com uma baixa eficiência da busca local. Isso é devido ao fato de quanto maior o valor do limitante menos o módulo de busca local é ativado realizando assim buscas em menos regiões do espaço de busca. Sendo assim, é preciso levar em consideração o tempo computacional gasto em cada configuração e a eficiência. Logo a abordagem com $\lambda = 10$ foi a mais competitiva obtendo bons resultados em pouco tempo computacional.

5.5.2 Resultados computacionais

Começamos a apresentação de nossos resultados, comparando a nossa solução com as soluções obtidas pela empresa. Na Tabela 5.9 mostramos as instâncias e o número de UMs consideradas seguidas pelo o custo da solução da empresa. Ao lado disso, mostramos o custo da solução do nosso algoritmo híbrido ALNS-CS (melhor e média de mais de 10 execuções), o tempo médio computacional e a melhoria percentual em relação à solução da empresa. Observamos que esse método supera consistentemente o da empresa, obtendo melhorias de até 32%.

Instância	n	Solução da empresa	**ALNS-CS**			
			Best	**Md**	**Tempo (s)**	**Melhoria(%))**
$UR-1$	18	2552.62	2521.92	2523.26	237	1.24
$UR-2$	18	3266.31	3183.01	3198.82	246	2.55
$UR-3$	19	4861.14	3290.19	3294.44	373	32.31
$UC-1$	41	6048.90	4345.54	4473.79	1721	28.15
$UC-2$	42	6295.16	4827.13	4995.72	1992	23.31
$UC-3$	40	6185.72	5035.43	5303.65	786	18.59
Média		4868.30	3867.20	3964.94	892	17.69

Tabela 5.9: Resumo de resultados, % melhoramento e tempo computacional para as instâncias UR e UC

Quando todas as UMs são consideradas em um único conjunto, a solução da empresa é simplesmente combinar ambas as soluções individuais. Na Tabela 5.10 mostramos como lidamos com as instâncias da classe UT. Esta tabela mostra o nome da instância seguido pelo número de UMs, a solução da empresa que é a soma das soluções individuais das instâncias usadas em sua construção. Em seguida, mostramos a nossa solução integrada considerando todas as UMs (melhores e médias), seguidas do tempo de execução e melhorias em relação à solução da empresa.

Instância	n	Solução da empresa	**ALNS-CS**			
			Best	**Md**	**Tempo (s)**	**Melhoria(%))**
$UT-1$	59	8602.52	6807.96	7007.41	2279	22.86
$UT-2$	60	9561.47	8070.34	8165.44	3968	17.27
$UT-3$	59	11046.86	8327.58	8562.73	3270	24.61
Média		9736.95	7735.29	7911.86	3172	20.91

Tabela 5.10: Resumo dos resultados,% melhoramento e tempo computacional para as instâncias UT

É óbvio que nossas soluções separadas da mesma maneira administrativa já eram melhores do que as da empresa, mas nossas soluções integradas podem encontrar

sinergias e melhorias ao programar e rotear embarcações, aprimorando ainda mais as soluções para este difícil e grande problema.

Na Figura 5.16 representamos a evolução do custo da solução em relação ao número de iterações, e observamos que, após cerca de 30 iterações, nosso algoritmo produziu uma solução melhor do que a da empresa. E esta solução ainda foi significativamente melhorada nas fases posteriores de nossa metaheurística.

Outro detalhe importante também pode ser destacado na Figura 5.16. Como foi visto na calibração da destruição de soluções do ALNS realizada na Seção 5.4.1, quanto maior a quantidade de clientes removidos, o tempo computacional cresce exponencialmente, entretanto a qualidade da solução aumenta já que gera uma maior análise na re-inserção desses clientes. A grande melhoria trazida pela busca local pode ser nitidamente observada nas quedas rápidas do valor da função objetivo em uma mesma iteração.

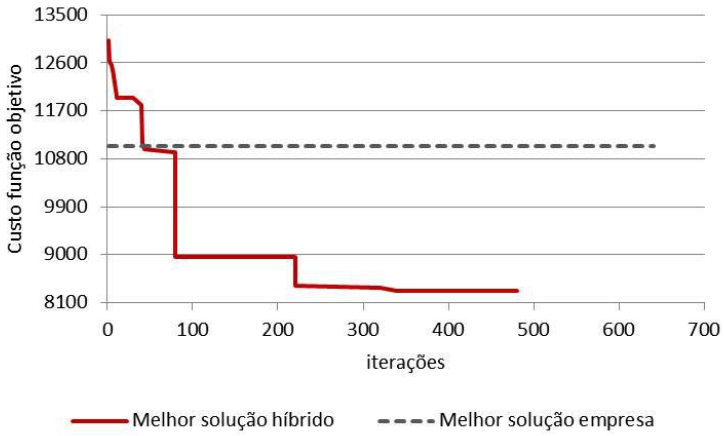


Figura 5.16: Evolução da melhores soluções encontradas pelo $UT - 3$

A inserção do método de assimilação PR, tem um grande impacto no início da heurística ao encontrar soluções melhores rapidamente. Esse método também se mostrou eficiente ao encontrar melhores soluções próximas às soluções encontradas pela busca local e pelo ALNS gerador quando realiza a conexão de caminhos entre a melhor solução encontrada e a anterior a ela, cobrindo assim uma lacuna deixada pelo ALNS. Na Figura 5.17 podemos ver quando o algoritmo encontra as melhores soluções através do PR destacando-se também onde são feitas as buscas locais (BL).

O módulo de busca local se apresentou como um importante componente, auxiliando na descoberta de melhores soluções de maneira eficiente. A ativação da busca local em grupos considerados promissores obteve sucesso em média 60% das vezes em que foi aplicado como podemos ver na Figura 5.18.

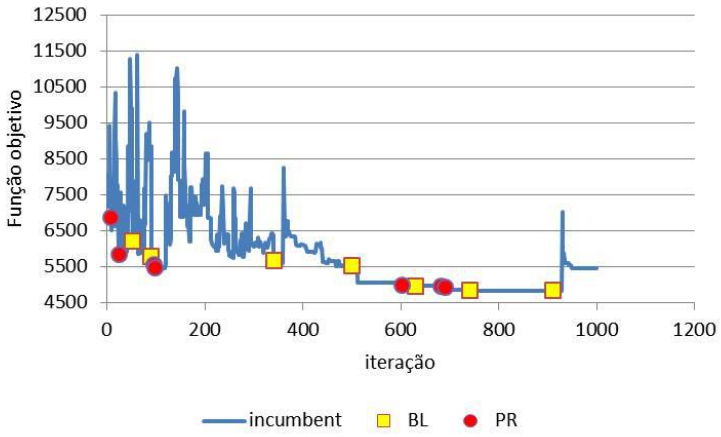


Figura 5.17: Melhores soluções encontradas pelo método de assimilação PR na instância $UC - 2$

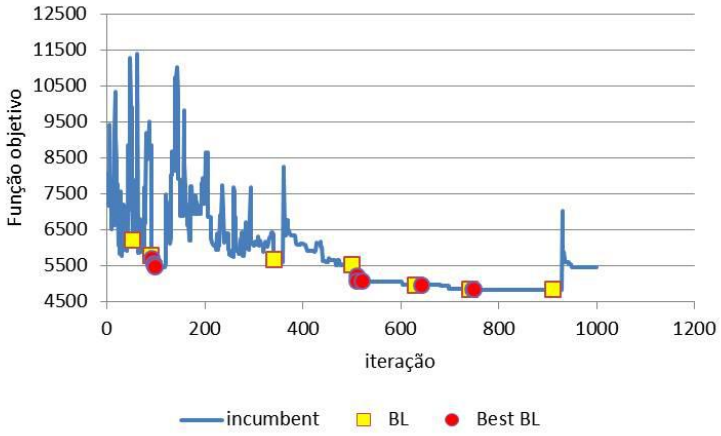


Figura 5.18: Eficiência do método de busca local para a instância $UC - 2$

A heurística ALNS que trabalha como módulo gerador encontra melhores soluções no início da heurística híbrida e com ao auxilio do módulo de busca local, ainda mostra bons resultados quando um grupo promissor é encontrado. A Figura 5.23 mostra as melhores soluções encontradas pela heurística geradora ALNS e ainda destaca as buscas locais mostrando que este módulo gerador continua sendo eficiente após entrar em grupos de soluções promissores.

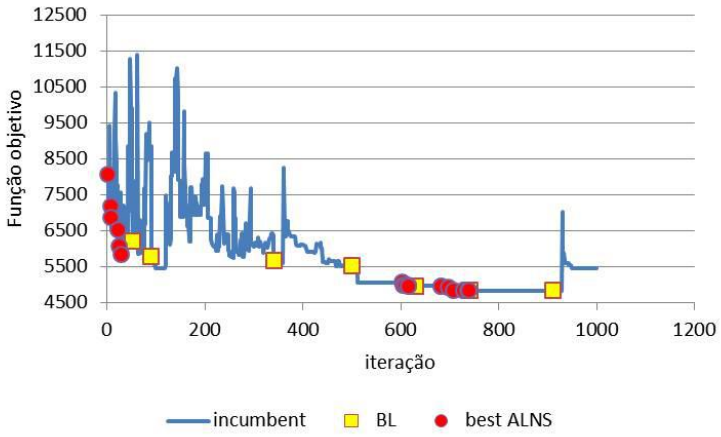


Figura 5.19: Melhores soluções encontradas pela heurística ALNS geradora para instância $UC - 2$

O método de múltiplas soluções inciais considerando uma porcentagem mínima de diversidade entre as soluções se mostrou eficiente ao ser capaz de diversificar o espaço de busca fazendo uma pertubação após uma busca local que não obteve sucesso. No gráfico da Figura 5.20 podemos observar como após a re-inserção de uma solução inicial o algoritmo escapa de um ótimo local encontrando melhores soluções.

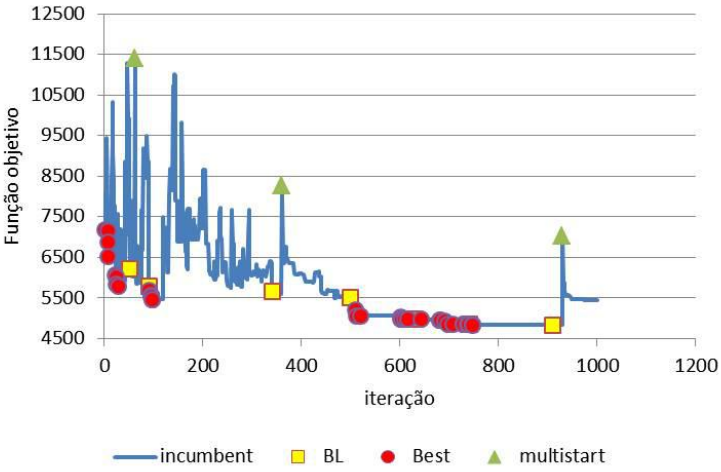


Figura 5.20: Eficiência da inserções de múltiplas soluções iniciais para a instância $UC - 2$

5.6 Comparação de resultados finais entre heurísticas

Nesta seção, avaliamos o desempenho dos algoritmos e a estabilidade em relação a alguns de seus parâmetros. Começamos apresentando o gráfico da Figura 5.21 com

as melhores soluções encontradas pelos métodos clusterização-roteamento (C-R), ALNS e híbrido comparados a solução da empresa em cada instância. A heurística híbrida supera claramente a solução da empresa assim como as outras heurísticas. A heurística ALNS mostra ter resultados concorrentes aos do método híbrido porém algumas análises a seguir comprovam a superioridade da hibridização ALNS-CS.

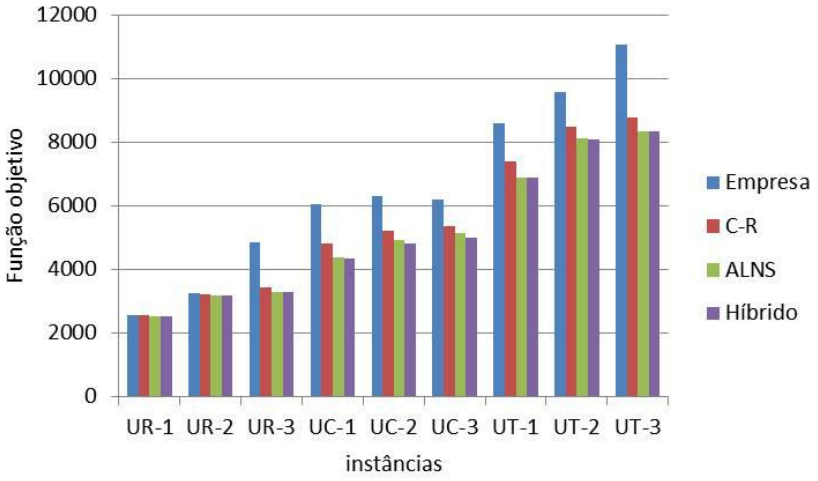


Figura 5.21: Comparação geral de diferentes abordagens

Como mostrado na Tabela 5.11, nossas soluções com a heurística ALNS e híbrida exigem menos embarcações do que as da empresa e da metodologia clusterização e roteamento (C-R). Isso é extremamente importante, uma vez que essas embarcações podem custar vários milhões de dólares, e ser capaz de realizar o serviço exigido com menos embarcações pode levar a empresa a uma economia significativa, ou ainda permitir que a empresa cresça e expanda suas operações sem grandes investimentos fixos em novas embarcações. Portanto, esses resultados mostram que a empresa deve avaliar sua forma de agrupar as UMs de maneira administrativa.

	# de viagens			
Instância	Empresa	C-R	ALNS	Híbrido
$UR-1$	4	4	3	3
$UR-2$	4	4	4	4
$UR-3$	6	5	4	4
$UC-1$	9	7	6	6
$UC-2$	9	7	7	7
$UC-3$	9	8	7	7
$UT-1$	13	10	9	9
$UT-2$	13	11	11	11
$UT-3$	13	12	11	11

Tabela 5.11: Comparativo de números de viagem por instância

Em seguida, a fim de comparar a heurística ALNS e o método híbrido, fazemos uma análise usando o coeficiente de variação, que representa a razão entre o desvio

padrão e a média *Dev = (desvio padrão / média)*, aplicando as heurísticas dez vezes para cada instância. Os resultados são apresentados na Tabela 5.12. Podemos ver que a hibridização proposta traz um grande avanço ao obter um coeficiente de variação mais baixo do que quando aplicada a heurística ALNS.

Instância	**ALNS** Dev(%)	**Híbrido** Dev(%)
$UR-1$	0.94	0.08
$UR-2$	0.50	0.47
$UR-3$	0.27	0.16
$UC-1$	4.28	2.60
$UC-2$	4.20	2.98
$UC-3$	3.75	3.06
$UT-1$	2.37	1.63
$UT-2$	1.76	0.66
$UT-3$	3.21	2.16
Média	2.36	1.53

Tabela 5.12: Coeficiente de variação

Outra análise que foi feita é a comparação dos métodos concorrentes ALNS e híbrido quanto ao tempo computacional gasto para encontrar as melhores soluções. O gráfico da Figura 5.22 mostra claramente a eficiência do método híbrido em relação ao tempo computacional para a instância UC-3. Isso ocorre devido as buscas locais nos grupos de soluções promissoras que fazem os custos da função objetivo descer rapidamente.

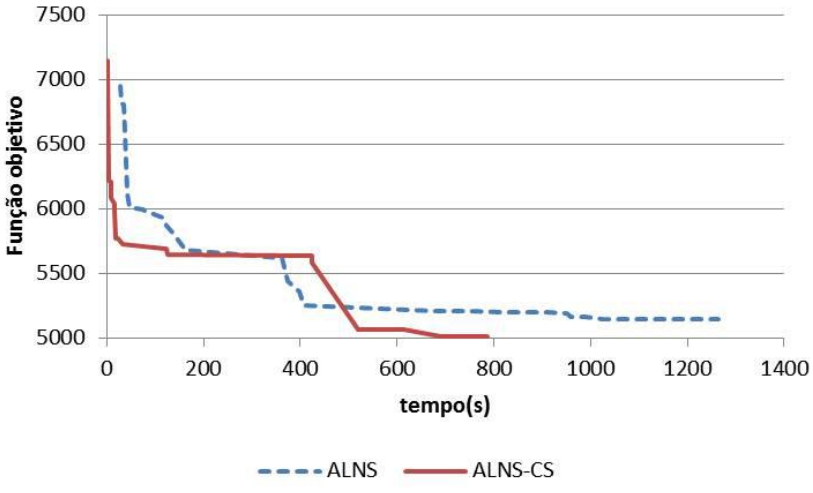


Figura 5.22: Evolução da melhores soluções encontradas pelos métodos ALNS e híbrido para a instância $UC-3$

Na Figura 5.23 vemos que o método híbrido com 1000 iterações (Híbrido-1000) encontra uma solução mais rapidamente e ainda continua encontrando melhores soluções até cerca de 1700 segundos enquanto que o ALNS (ALNS-1000) permanece

na mesma região de busca. Quando aumentamos o número de iterações para 1500, cedendo mais tempo computacional total de busca para ambos os métodos, o ALNS leva ainda mais tempo para encontrar a melhor solução.

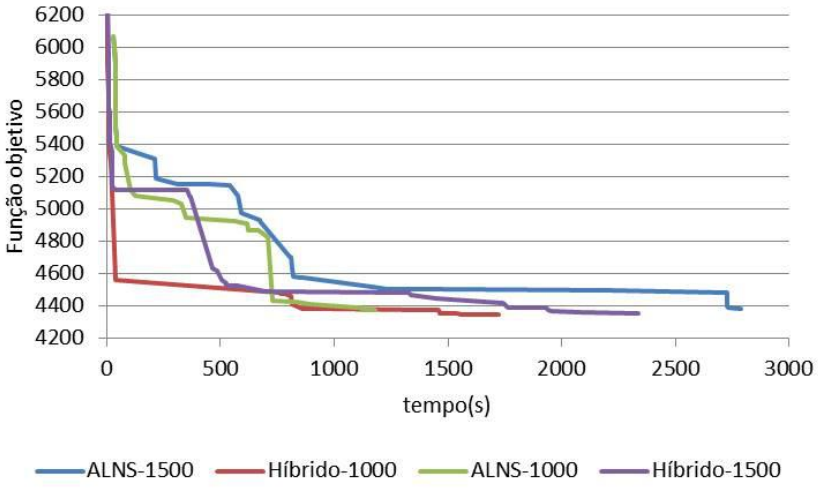
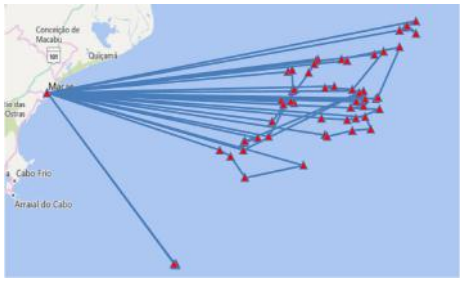
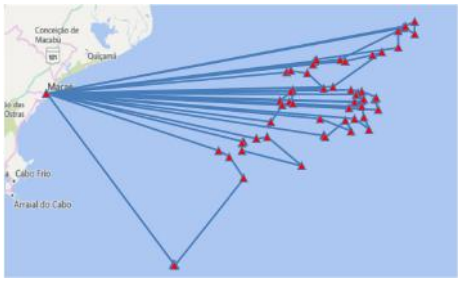


Figura 5.23: Diferença entre as melhores soluções encontradas em função do tempo pelo ALNS e híbrido aumentando o número de iterações na instância $UC - 2$

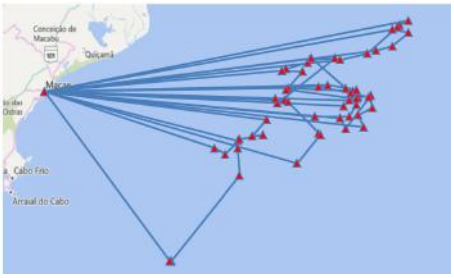
Finalmente, mostramos uma representação gráfica das rotas ao longo da costa do Brasil para a instância UT-3 que apresentou a maior diferença entre melhores soluções. Na Figura 5.24(a), mostramos as rotas da solução da empresa, na Figura 5.24 (b) as rotas formadas pelo método de clusterização-roteamento, na Figura 5.24 (c) as rotas da heurística ALNS e na Figura 5.24 (d) as da heurística híbrida ALNS-CS. Pode-se ver que a mudança na forma da solução é significativa, reforçando novamente a ideia de que nossa solução melhorou não apenas o aspecto das rotas, mas também a programação e agrupamento de UMs em cada rota.



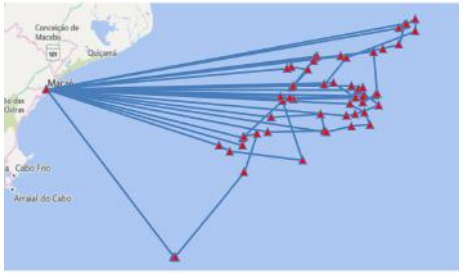
(a) Rotas da empresa (custo da solução = 11046.86)



(b) Rotas da clusterização-roteamento (custo da solução = 8786.56)



(c) Rotas da heurística ALNS (custo da solução =8331.08)



(d) Rotas da heurística híbrida ALNS-CS (custo da solução =8327.58)

Figura 5.24: Mapa de rotas

Capítulo 6

Conclusões

Nos últimas anos, problemas de otimização combinatória tem sido resolvidos de forma inteligente em um tempo razoável e com soluções de boa qualidade por uma ferramenta interessante, as metaheurísticas. O conceito de hibridização entre heurísticas também vem ganhando cada vez mais importância uma vez que conhecimentos específicos sobre o problema e o espaço de busca podem ser determinantes para alcançar uma melhor performance na resolução de problemas difíceis.

Em geral as metaheurísticas tem o objetivo de obter um equilíbrio entre diversificação e intensificação, evitando assim convergência prematura para regiões ruins e possibilitando identificar boas regiões para então efetuar a busca e obter sucesso. Visto que a utilização do método exato para resolver o problema pode possuir um alto tempo computacional ou nem encontrar soluções viáveis, a aplicação de busca local somente em regiões consideradas promissoras revela-se uma técnica interessante.

Uma forma de detectar regiões promissoras é por meio da identificação da quantidade de soluções geradas em cada região do espaço de busca. Então estabelece-se um critério em que regiões nas quais muitas soluções são geradas tendem a ser regiões que contenham melhores soluções.

Este trabalho discutiu e modelou um método híbrido baseado na heurística ALNS que gera boas soluções e o método CS que divide o espaço de busca em subespaços e detecta regiões promissoras por mérito de frequência. A busca é intensificada nas regiões promissoras tão logo estas sejam detectadas por meio do módulo de análise.

Essa abordagem híbrida foi aplicada ao PRESP que lida com instâncias baseadas em dados reais da companhia de petróleo brasileira parceira. O PRESP é um problema que possui uma dificuldade maior que outros problemas de PRV pois operam sob diferentes condições e e exigem modelos de suporte à decisão que se encaixam nas características do problema específico. Além disso, no caso de uma aplicação em instâncias reais, a redução com custos de transporte é imprescindível para alcançar um excelente plano logístico. Por isso é importante que o método

de resolução consiga levar em consideração todos os detalhes do problema e principalmente seja capaz de resolvê-lo em um tempo considerado viável para os que planejam o processo de transporte marítimo.

As principais contribuições deste trabalho estão relacionas as melhorias propostas para a heurística ALNS através da inserção de detecção de áreas promissoras no espaço de busca, tornando-o mais inteligente ao realizar busca local. Para isso, utiliza-se técnicas do método CS onde foram adicionados novos componentes ao ALNS que lidam com múltiplas soluções iniciais, assimilação de caminhos, análise de grupos de soluções e eficiência da busca local. É focado também na importância de dividir os grupos e as soluções de re-inicialização com uma diversidade que garantisse uma maior cobertura do espaço de busca.

Uma das metas do trabalho era mostrar que essa hibridização torna a busca mais eficiente e racional. Resultados positivos foram encontrados quando comparamos as soluções da heurística ALNS original e da híbrida verificando uma redução considerável no coeficiente de variação entre as soluções geradas com diferentes sementes com médias melhores, assim como melhores soluções. Podemos concluir que uma heurística que acrescenta módulos de busca, mapeando de forma inteligente obtém a capacidade de escapar de ótimos locais e encontrar melhores soluções diferentemente do ALNS que permanece na mesma região.

Além disso, adicionamos técnicas que não são encontradas no CS original como o PR dentro do módulo de busca local e dentro do módulo gerador de soluções, que neste caso é o ALNS. Concluímos que o PR aplicado dentro desses módulos tem um grande impacto no início da heurística encontrando melhores soluções rapidamente, assim como ao aperfeiçoar as soluções encontradas no módulo BL e módulo ALNS gerador.

A fim de não gerar um *loop* em torno de um ótimo local fazendo com que o centro do grupo de soluções sempre retorne a mesma solução, utiliza-se uma variável que mede a eficiência do componente de busca local quando ativado. Esta variável chamada de índice de ineficácia (r_i) permite controlar se uma região é ruim ou se já foi suficientemente explorada. Quando esse índice é alto, uma pertubação é aplicada através da inserção de outra solução inicial com diversidade controlada. Isso permite que a heurística híbrida escape de um ótimo local possibilitando a pesquisa em uma região de busca diferente.

Este trabalho também contribuiu ao mostrar diferentes metodologias capazes de resolver o PRESP de forma inteligente utilizando uma abordagem de três fases que combina uma heurística de agrupamento e um método exato, outra abordagem que lida com a heurística ALNS e por fim a hibridização entre heurísticas ALNS e CS que gera as melhores soluções.

Os resultados em geral mostraram uma grande redução dos custos da função

objetivo se comparada a técnica utilizada pela empresa. A metodologia que utiliza três fases apesar de reduzir os custos da função objetivo em cerca de 16%, se mostrou mais trabalhosa e adaptada a programação do porto ao considerar as partidas fixas. Além disso, por utilizar o método exato em duas das três fases do método, possui um maior tempo computacional com soluções piores que as encontradas pelas heurísticas posteriores. Já a aplicação da heurística ALNS mostrou-se eficiente ao encontrar soluções melhores com uma redução de cerca de 18% se comparada a solução da empresa e com um tempo computacional consideravelmente menor. Entretanto, essa heurística se mostra pouco eficaz quando precisa sair de ótimos locais gerando soluções com um alto coeficiente de variação.

A última abordagem proposta para essa tese foi a heurística híbrida ALNS-CS que além de encontrar soluções melhores, reduz a média de melhores soluções encontradas assim como o coeficiente de variação trazendo uma maior confiabilidade ao processo. Isso é devido a sua capacidade de mapeamento do espaço de busca através da divisão em grupos de soluções e por consequência se torna capaz de identificar regiões ruins no espaço de busca e escapar de ótimos locais. A redução do custo da função objetivo fica em média 19% mais baixo que o da empresa e acima das reduções encontradas anteriormente.

Por fim, outra contribuição importante para este trabalho é a realização de diversos testes que permitiram compreender melhor o funcionamento tanto do ALNS como dos módulos implementados na heurística híbrida. A análise do comportamento do método perante a vários fatores que podem influenciar no processo de busca traz uma compreensão mais clara do seu funcionamento.

Por meio desses testes podemos ver que no ALNS, se a temperatura inicial for considerada suficientemente alta, não altera em grande escala a média de soluções; a porcentagem de destruição das soluções com os operadores de remoção tem um grande potencial de melhoria se for aumentada, contudo aplicada frequentemente aumenta exponencialmente o tempo computacional; e finalmente o aumento do número de iterações aumenta as chances de obter melhores soluções sendo necessário considerar o aumento do tempo computacional.

Já a heurística híbrida teve os parâmetros concernentes as técnicas do CS ajustadas e o número de grupos de soluções e a distância entre eles influencia na eficiência da busca local; o estudo dos diferentes valores dos limitantes λ propostos mostram que o tempo computacional diminui a medida que aumenta-se o seu valor, porém a eficiência da busca local local diminui por não conseguir investigar centros suficientes.

Portanto a hibridização proposta para esse trabalho associado a modificação de algumas técnicas originais do ALNS e do CS trazem maior clareza em relação as diferentes formas de busca e mostra a importância da aplicação de heurísticas

voltadas para a resolução de problemas reais. Sendo assim, acredita-se que este trabalho seja uma boa fonte de contribuição para trabalhos futuros, tanto a critério de melhoramento como de comparação.

Como ideias para trabalhos futuros na área que utiliza o método das três fases sugerimos talvez a utilização de uma outra heurística agrupamento mais complexa que leve em conta, por exemplo, outras características do problema como quais unidades possuem restrições de atendimento. Já para o ALNS outros tipos de operadores de remoção e inserção poderiam alcançar melhores resultados.

Apesar de apresentar bons resultados na hibridização algumas melhorias ainda podem ser aplicadas. Na parte teórica testes podem ser feitos considerando, por exemplo, uma métrica de distância diferente para definir o grupo de soluções mais similar, outra forma de verificar se o grupo é promissor ou ainda quanto a formação de grupos iniciais.

Outras heurísticas de busca local podem ser aplicadas ao invés da utilização do próprio ALNS modificado ou ainda considerar a utilização de outros operadores. A paralelização do método gerador de soluções ou dos componentes seriam alternativas interessantes para instâncias maiores.

Referências Bibliográficas

[1] LEITE, R. P. "Maritime transport of deck cargo to Petrobras fields in Campos Basin: An empirical analysis, identification and quantification of improvement points", *Departamento de Engenharia Industrial, Pontifıcia Universidade Catolica do Rio de Janeiro, MSc Thesis, Rio de Janeiro, Brazil*, 2012.

[2] RODRIGUE, J.-P. *The Geography of Transport Systems*. New York: Routledge, 2017.

[3] CHRISTIANSEN, M., FAGERHOLT, K., RONEN, D. "Ship routing and scheduling: Status and perspectives", *Transportation Science*, v. 38, n. 1, pp. 1–18, 2004.

[4] FREI, C. "Global and regional issues: The energy challenges for the future", *Energy Leaders' Summit, Intanbul*, 2012.

[5] FISCHETTI, M., TOTH, P. "An additive approach for the optimal solution of the prize collecting traveling salesman problem", *Vehicle Routing: Methods and Studies*, pp. 319–343, 1988.

[6] FERREIRA FILHO, V. J. M., BAHIENSE, L., DE ARRUDA, E. F., et al. *Projeto Programação & Dimensionamento de Frota Relatório Técnico Final*. Relatório técnico, Petrobras, 2015.

[7] PISINGER, D., ROPKE, S. "A general heuristic for vehicle routing problems", *Computers & Operations Research*, v. 34, n. 8, pp. 2403–2435, 2007.

[8] BOUCHER, J., COELHO, L., FERREIRA FILHO, V., et al. "Cyclic Maritime Routing of Plattaform Supply Vessels", *Computers & Operations Research*, 2018. Artigo submetido em abril de 2018.

[9] OLIVEIRA, A., LORENA, L. A. "Detecting promising areas by evolutionary clustering search". In: *Brazilian Symposium on Artificial Intelligence*, pp. 385–394. Springer, 2004.

[10] NOVAES, A. *Logística e gerenciamento da cadeia de distribuição*. Elsevier Brasil, 2016.

[11] SONG, D.-W., PANAYIDES, P. *Maritime Logistics: A Guide to Contemporary Shipping and Port Management*. Kogan Page, 2015.

[12] LABADIE, N., PRINS, C. "Vehicle routing nowadays: Compact review and emerging problems". In: *Production systems and supply chain management in emerging countries: best practices*, Springer, pp. 141–166, 2012.

[13] KAISER, M., SNYDER, B. "An empirical analysis of offshore service vessel utilization in the US Gulf of Mexico", *International Journal of Energy Sector Management*, v. 4, n. 2, pp. 152–182, 2010.

[14] NISHIMURA, E., IMAI, A., PAPADIMITRIOU, S. "Berth allocation planning in the public berth system by genetic algorithms", *European Journal of Operational Research*, v. 131, n. 2, pp. 282–292, 2001.

[15] HERMETO, N., FERREIRA FILHO, V., BAHIENSE, L. "Logistics network planning for offshore air transport of oil rig crews", *Computers & Industrial Engineering*, v. 75, pp. 41–54, 2014.

[16] SOLOMON, M. "Algorithms for the vehicle routing and scheduling problems with time window constraints", *Operations Research*, v. 35, n. 2, pp. 254–265, 1987.

[17] CORDEAU, J.-F., GENDREAU, M., LAPORTE, G. "A tabu search heuristic for periodic and multi-depot vehicle routing problems", *Networks*, v. 30, n. 2, pp. 105–119, 1997.

[18] FRANCIS, P., SMILOWITZ, K., TZUR, M. "The period vehicle routing problem and its extensions". In: *The vehicle routing problem: latest advances and new challenges*, Springer, pp. 73–102, 2008.

[19] VIDAL, T., CRAINIC, T., GENDREAU, M., et al. "A hybrid genetic algorithm for multidepot and periodic vehicle routing problems", *Operations Research*, v. 60, n. 3, pp. 611–624, 2012.

[20] BALDACCI, R., BARTOLINI, E., MINGOZZI, A., et al. "An exact algorithm for the period routing problem", *Operations research*, v. 59, n. 1, pp. 228–241, 2011.

[21] BELTRAMI, E., BODIN, L. "Networks and vehicle routing for municipal waste collection", *Networks*, v. 4, n. 1, pp. 65–94, 1974.

[22] RUSSELL, R., IGO, W. "An assignment routing problem", *Networks*, v. 9, n. 1, pp. 1–17, 1979.

[23] CHRISTOFIDES, N., BEASLEY, J. "The period routing problem", *Networks*, v. 14, n. 2, pp. 237–256, 1984.

[24] TAN, C., BEASLEY, J. "A heuristic algorithm for the period vehicle routing problem", *Omega*, v. 12, n. 5, pp. 497–504, 1984.

[25] RUSSELL, R., GRIBBIN, D. "A multiphase approach to the period routing problem", *Networks*, v. 21, n. 7, pp. 747–765, 1991.

[26] LAPORTE, G. "Fifty years of vehicle routing", *Transportation Science*, v. 43, n. 4, pp. 408–416, 2009.

[27] CHAO, I., GOLDEN, B., WASIL, E., et al. "An improved heuristic for the period vehicle routing problem", *Networks*, v. 26, n. 1, pp. 25–44, 1995.

[28] FRANCIS, P., SMILOWITZ, K., TZUR, M. "The period vehicle routing problem with service choice", *Transportation Science*, v. 40, n. 4, pp. 439–454, 2006.

[29] MOURGAYA, M., VANDERBECK, F. "Column generation based heuristic for tactical planning in multi-period vehicle routing", *European Journal of Operational Research*, v. 183, n. 3, pp. 1028–1041, 2007.

[30] GAUDIOSO, M., PALETTA, G. "A heuristic for the periodic vehicle routing problem", *Transportation Science*, v. 26, n. 2, pp. 86–92, 1992.

[31] NGUYEN, P. K., CRAINIC, T. G., TOULOUSE, M. *A hybrid genetic algorithm for the periodic vehicle routing problem with time windows*. CIRRELT, 2011.

[32] DRUMMOND, L., OCHI, L., VIANNA, D. "An asynchronous parallel metaheuristic for the period vehicle routing problem", *Future generation computer systems*, v. 17, n. 4, pp. 379–386, 2001.

[33] FAGERHOLT, K., LINDSTAD, H. "Optimal policies for maintaining a supply service in the Norwegian Sea", *Omega*, v. 28, n. 3, pp. 269–275, 2000.

[34] HALVORSEN-WEARE, E., FAGERHOLT, K., NONÅS, L., et al. "Optimal fleet composition and periodic routing of offshore supply vessels", *European Journal of Operational Research*, v. 223, n. 2, pp. 508 – 517, 2012.

[35] HALVORSEN-WEARE, E. E., FAGERHOLT, K. "Optimization in offshore supply vessel planning", *Optimization and Engineering*, v. 18, n. 1, pp. 317–341, 2017.

[36] SHYSHOU, A., GRIBKOVSKAIA, I., LAPORTE, G., et al. "A large neighbourhood search heuristic for a periodic supply vessel planning problem arising in offshore oil and gas operations", *INFOR: Information Systems and Operational Research*, v. 50, n. 4, pp. 195–204, 2012.

[37] ROPKE, S., PISINGER, D. "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows", *Transportation Science*, v. 40, n. 4, pp. 455–472, 2006.

[38] BORTHEN, T., LOENNECHEN, H., WANG, X., et al. "A genetic search-based heuristic for a fleet size and periodic routing problem with application to offshore supply planning", *EURO Journal on Transportation and Logistics*, pp. 1–30, 2017.

[39] SHAW, P. "Using constraint programming and local search methods to solve vehicle routing problems". In: *Principles and Practice of Constraint Programming*, Springer, pp. 417–431, 1998.

[40] KISIALIOU, Y., GRIBKOVSKAIA, I., LAPORTE, G. "The periodic supply vessel planning problem with flexible departure times and coupled vessels", *Computers & Operations Research*, 2018.

[41] SALAZAR-AGUILAR, M., LANGEVIN, A., LAPORTE, G. "Synchronized arc routing for snow plowing operations", *Computers & Operations Research*, v. 39, n. 7, pp. 1432–1440, 2012.

[42] RIBEIRO, G. M., DESAULNIERS, G., DESROSIERS, J., et al. "Efficient heuristics for the workover rig routing problem with a heterogeneous fleet and a finite horizon", *Journal of Heuristics*, v. 20, n. 6, pp. 677–708, 2014.

[43] COELHO, L., CORDEAU, J.-F., LAPORTE, G. "Consistency in multi-vehicle inventory-routing", *Transportation Research Part C: Emerging Technologies*, v. 24, pp. 270–287, 2012.

[44] RIBEIRO, G., LAPORTE, G. "An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem", *Computers & operations research*, v. 39, n. 3, pp. 728–735, 2012.

[45] GHILAS, V., DEMIR, E., VAN WOENSEL, T. "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time

windows and scheduled lines", *Computers & Operations Research*, v. 72, pp. 12–30, 2016.

[46] MULLER, L., SPOORENDONK, S. *A hybrid adaptive large neighborhood search algorithm applied to a lot-sizing problem.* Relatório técnico, DTU Management, 2010.

[47] KOÇ, Ç., BEKTAŞ, T., JABALI, O., et al. "A hybrid evolutionary algorithm for heterogeneous fleet vehicle routing problems with time windows", *Computers & Operations Research*, v. 64, pp. 11–27, 2015.

[48] OLIVEIRA, A., LORENA, L. "Hybrid evolutionary algorithms and clustering search". In: *Hybrid Evolutionary Algorithms*, Springer, pp. 77–99, 2007.

[49] CORRÊA, F., CHAVES, A.A.AND LORENA, L. . N. "Hybrid heuristics for the probabilistic maximal covering location-allocation problem", *Operational Research*, v. 7, n. 3, pp. 323–343, 2007.

[50] BIAJOLI, F. L., LORENA, L. A. N. "Clustering search approach for the traveling tournament problem". In: *Mexican international conference on artificial intelligence*, pp. 83–93. Springer, 2007.

[51] OLIVEIRA, R. M., MAURI, G. R., LORENA, L. A. N. "Clustering search for the berth allocation problem", *Expert Systems with Applications*, v. 39, n. 5, pp. 5499–5505, 2012.

[52] RIBEIRO, G., LAPORTE, G., MAURI, G. "A comparison of three metaheuristics for the workover rig routing problem", *European Journal of Operational Research*, v. 220, n. 1, pp. 28–36, 2012.

[53] REUTERS. "Offshore rig driller seadrill hopest to see market turn in 2017", *Reporting by Ole Petter Skonnord, editing by David Evans*, 2015. Disponível em: <http://www.rigzone.com/news/article.asp?a_id=141009>.

[54] FRIEDBERG, D. O., UGLANE, V. T. "Routing and Scheduling of Platform Supply Vessels", *Norwegian University of Science and Technology*, 2013.

[55] NEMHAUSER, G. L., WOLSEY, L. A. *Integer and Combinatorial Optimization.* New York, NY, USA, Wiley-Interscience, 1988. ISBN: 0-471-82819-X.

[56] GAREY, M. R., JOHNSON, D. S. *Computers and Intractability; A Guide to the Theory of NP-Completeness.* New York, NY, USA, W. H. Freeman & Co., 1990. ISBN: 0716710455.

[57] BLUM, C., ROLI, A. "Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison", *ACM Comput. Surv.*, v. 35, n. 3, pp. 268–308, set. 2003. ISSN: 0360-0300. doi: 10.1145/937503.937505. Disponível em: <http://doi.acm.org/10.1145/937503.937505>.

[58] CHRISTIANSEN, M., FAGERHOLT, K., NYGREEN, B., et al. "Maritime transportation", *Handbooks in Operations Research and Management Science*, v. 14, pp. 189–284, 2007.

[59] EIBEN, A. E., SCHIPPERS, C. A. "On Evolutionary Exploration and Exploitation", *Fundam. Inf.*, v. 35, n. 1-4, pp. 35–50, ago. 1998. ISSN: 0169-2968. Disponível em: <http://dl.acm.org/citation.cfm?id=297119.297124>.

[60] SCHRIMPF, G., SCHNEIDER, J., STAMM-WILBRANDT, H., et al. "Record breaking optimization results using the ruin and recreate principle", *Journal of Computational Physics*, v. 159, n. 2, pp. 139–171, 2000.

[61] DEES JR, W. A., KARGER, P. G. "Automated rip-up and reroute techniques". In: *Proceedings of the 19th Design Automation Conference*, pp. 432–439. IEEE Press, 1982.

[62] AHUJA, R. K., ERGUN, Ö., ORLIN, J. B., et al. "A survey of very large-scale neighborhood search techniques", *Discrete Applied Mathematics*, v. 123, n. 1, pp. 75–102, 2002.

[63] HANSEN, P., MLADENOVIĆ, N. "An Introduction to Variable Neighborhood Search". In: *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, pp. 433–458, Boston, MA, Springer US, 1999.

[64] METROPOLIS, N., ROSENBLUTH, A. W., ROSENBLUTH, M. N., et al. "Equation of state calculations by fast computing machines", *The journal of chemical physics*, v. 21, n. 6, pp. 1087–1092, 1953.

[65] GILLETT, B., MILLER, L. R. "A heuristic algorithm for the vehicle-dispatch problem", *Operations Research*, v. 22, n. 2, pp. 340–349, 1974.

[66] HAMMING, R. W. "Error detecting and error correcting codes", *Bell Labs Technical Journal*, v. 29, n. 2, pp. 147–160, 1950.

[67] SILVA, G. C. D., OCHI, L. S., MARTINS, S. L. "Proposta e avaliação de heurísticas grasp para o problema da diversidade máxima", *Pesquisa Operacional*, v. 26, n. 2, pp. 321–360, 2006.

[68] KUO, C.-C., GLOVER, F., DHIR, K. "Analyzing and Modeling the Maximum Diversity Problem by Zero-One Programming", *Decision Sciences*, v. 24, n. 6, pp. 1171–1185, 1993.

[69] GLOVER, F., LAGUNA, M. *Tabu Search*. Norwell, MA, USA, Kluwer Academic Publishers, 1997. ISBN: 079239965X.