Mini-curso: Introdução à otimização de operações portuárias

A.T. Azevedo*, A.A. Chaves, L.L. Salles Neto

12 de agosto de 2014

Resumo: Neste texto são apresentados os tipos existentes de portos quanto à carga movimentada e a motivação apropriada para estudar, em particular, os portos com movimentação de contêineres. São identificadas os cinco principais problemas existentes neste tipo de porto e três deles são abordados do ponto de vista de modelo matemático e método de solução. Todos os três problemas são NP-Completo e algumas das técnicas de solução empregadas e comparadas foram: algoritmo genético com chaves aleatórias viciadas, *Clustering Search*, *Beam Search*, algoritmo genético com representação por regras e *Simulated Annealing*.

Palavras-chave: Operações portuárias, alocação de berços, plano de estiva, *scheduling* de guindastes portuários, Algoritmo genético com chaves aleatórias viciadas, *Clustering Search*, *Beam Search*, Algoritmo genético, *Simulated Annealing*.

Abstract: *This text presents the types of port according to operated cargo and why it is important to study ports which cargo are containers. Five main problems are defined and three of them are treated through a mathematical model and proper solutions methods. All three problems are NP-Complete and some solution techniques employed and compared: biased-random key genetic algorithm, clustering search, beam search, genetic algorithm coupled with representation by rules, and simulated annealing.*

Keywords: *Ports operations, berth allocation, stowage planning, Quay Crane Scheduling, biased-random key genetic algorithm, clustering search, beam search, genetic algorithm coupled with representation by rules, and simulated annealing.*

*Autor para contato: anibal.azevedo@fca.unicamp.br

Prefácio

A ideia desse mini-curso nasceu durante o desenvolvimento do projeto de inovação tecnológica "Otimização das operações de carregamento e descarregamento de navios", um dos projetos financiados pela empresa Vale e pela Fapesp no âmbito do programa Projeto de Inovação Tecnológica na Empresa (PITE).

Para o desenvolvimento do projeto foi preciso compreender o problema na prática, em especial no Porto de Tubarão, operado pela empresa Vale em Vitória (ES), para assim viabilizar a formulação de modelos, métodos e sistemas computacionais. Um dos sistemas desenvolvidos teve um pedido de patente recentemente depositado no INPI (Sistema para otimizar a relação entre custos de carregamento de navios e transporte de cargas em navios atracados - Registro BR1020140157492). A implantação desse sistema na operação de um dos terminais do Porto de Tubarão está sendo encaminhada nesse momento pela empresa.

I Alguns métodos desenvolvidos pelos autores estão sendo apresentados nesse simpósio, e outros ainda estão sendo confeccionados e finalizados. Contudo, o grande número de problemas complexos presentes numa operação portuária motivou os autores a proporem esse mini-curso visando, de forma introdutória, apresentar os principais problemas de otimização presentes em operações portuárias e alguns métodos de solução. Acreditamos que se trata de uma pequena contribuição na fomentação de novos grupos de pesquisa com essa temática no país.

Somos muito gratos à Sobrapo pela oportunidade de ministrar esse mini-curso durante o XLVI Simpósio Brasileiro de Pesquisa Operacional.

Aníbal Tavares de Azevedo

Antonio Augusto Chaves

Luiz Leduíno de Salles Neto

São José dos Campos, 12 de agosto de 2014

1 Introdução

No Brasil o setor portuário é responsável por 95% do volume do comércio exterior. Se mantidas as taxas de crescimento de 5% dos últimos anos, os portos brasileiros terão que aumentar sua capacidade de atendimento de 650 milhões de toneladas por ano, em 2012, para 900 milhões em 2017. Esse acréscimo de 40%, ou 32 milhões de toneladas por ano, equivale a um porto de Santos a cada três anos.

É notória, assim, a importância estratégica da otimização das operações portuárias no Brasil. Reportagem publicada no jornal O Estado de São Paulo em 30 de setembro de 2012 afirma que:

> "O modelo de gestão dos portos é confuso, difícil de ser gerenciado e representa um gargalo para o Brasil, na avaliação do governo, que prepara um pacote que promete uma "revolução" no setor. (...)"

Já a reportagem publicada na revista Exame em 5 de outubro de 2011, intitulada "Os portos brasileiros são um desastre", traz um relato muito interessante e ilustrativo (Vettorazzo, 2011):

> *"O toque do celular do carioca Rafael Malafaia - um ringtone com uma guitarra estridente - está sempre regulado no volume máximo. O objetivo é ser acordado a qualquer hora do dia ou da noite quando surge algum problema de trabalho. Malafaia é o responsável pelo planejamento das cargas transportadas pelo navio Jacarandá, da empresa de logística LogIn, com sede no Rio de Janeiro. É ele que determina o local onde cada contêiner deve ser posicionado dentro do navio, de forma que, na hora do desembarque, a operação seja a mais rápida possível. Trata-se de uma tarefa complicada, pois o Jacarandá carrega até 2800 contêineres de 20 toneladas cada um. Na madrugada de 24 de agosto, o celular de Malafaia não parou de tocar. Ele coordenava, de seu quarto no Jacarandá, a operação de partida do porto de Santos, o maior do país. A barulheira só cessou às 4h38 da manhã, quando o navio saiu de Santos rumo a Paranaguá, no Paraná. Irritado e com os olhos vidrados na tela do computador, Malafaia desabafou: "Droga! Deixamos muita carga no chão", referindo-se aos 142 contêineres deixados para trás, o equivalente a 30% da carga que deveria ter sido coletada no porto paulista".*

Nota-se nesse relato, além do grande desperdício, a ausência de um processo inteligente de carregamento do navio, que parece depender exclusivamente da experiência de um operador, que não deve ser descartada, mas sim melhor aproveitada. Infelizmente esse relato não parece ser uma exceção nos portos brasileiros.

Esse mini-curso objetiva apresentar os principais problemas enfrentados por grande portos brasileiros, seus modelos matemáticos, bem como mostrar alguns métodos de resolução. Serão abordados três problemas em especial: o problema de alocação de navios em berços, o problema de carregamento e descarregamento de navios porta-contêineres (plano de estiva), e o *Scheduling* de guindastes portuários em conjunto com o plano de estiva.

Antes de abordar estes problemas, os modelos matemáticos correspondentes e alguns métodos de solução, é necessário entender os tipos de portos, e o funcionamento de um tipo em particular: o porto com movimentação de contêineres.

De acordo com Ligteringen & Velsink (2012) é possível classificar os terminais portuários, segundo a carga movimentada, em quatro principais tipos [1]:

- Contêiner: associado a movimentação de produtos armazenados dentro de contêineres.
- *Dry bulk* ou Granel sólido: produtos à granel como soja, fertilizantes, carvão, e minério de ferro.
- *Liquid Bulk* ou Granel líquido: petróleo e seus derivadas, gás na forma liquefeita, por exemplo, são produtos movimentados neste tipo de terminal.
- Carga geral e terminais multi-propósito: pode movimentar produtos dentro e fora de contêineres, como por exemplo, carros. Ou ainda, conjugar atividades referentes a dois mais tipos de produtos diferentes (Granel sólido ou líquido).

Exemplos dos quatro tipos de terminais são dados nas Figuras 1.

(a) Contêiner (Fonte:Port (2014))

(b) Granel sólido (Fonte:Terminals (2014a))

(c) Granel líquido (Fonte:Terminals (2014b))

(d) Carga geral (Fonte:Terminal (2014))

Figura 1: Ilustração de tipos de terminais quanto a carga movimentada.

Os terminais de carga à granel sólido ou líquido não serão objeto de discussão neste texto, mas são indicados as seguintes referências para o leitor interessado: roteamento e programação de navios para granel líquido (Al-Khayyal & Hwang, 2007; Bausch et al., 1998; Christiansen et al., 2004); para o problema geral de dimensionamento de frota de navios são indicadas as referências (Perakis & Jaramillo, 1991; Jaramillo & Perakis, 1991; Powell & Perkins, 1997); carga à granel é necessário considerar algumas características da carga na hora do navio

[1] existem ainda terminais para movimentação de frutas, pescado, veículos com rodas, passageiros (de curta distância ou "ferry boats", e de cruzeiros).

atracar no porto (Barros et al., 2011) ou ainda de como empregar simulação para saber onde pode estar o gargalho operativo do porto (Cassettari et al., 2012).

A Figura 2 ilustra a complexidade de se planejar um porto cujo produto a ser recebido é carvão para realizar o abastecimento de uma unidade termelétrica.

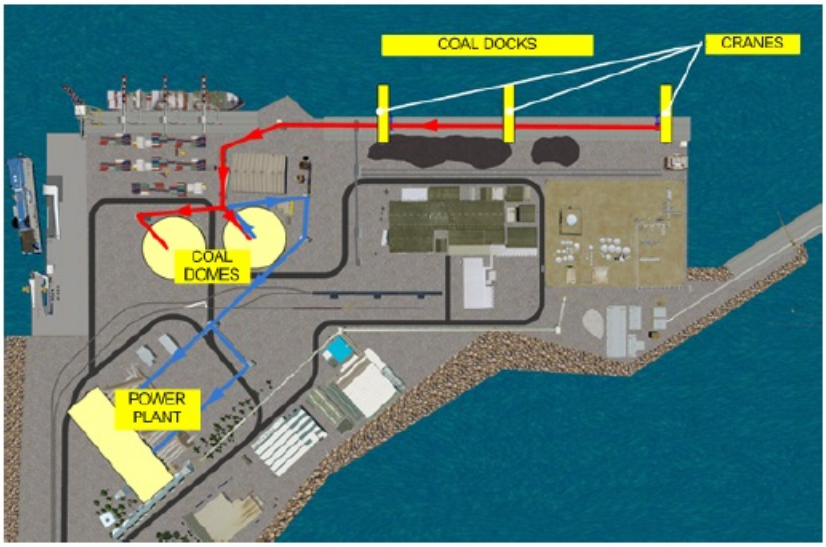


Figura 2: Elementos de um porto de granel sólido: carvão. Fonte: Cassettari et al. (2012).

Antes de detalhar o funcionamento de um porto de contêineres, é interessante investigar a evolução do transporte de carga por meio de contêineres, bem como o crescimento de sua importância no comércio internacional. O uso de contêineres no mercado de transporte marítimo começou há quase seis décadas. Antes disso, toda carga era manuseada de forma manual nos portos. Navios cargueiros permaneciam atracados durante vários dias ou até semanas para que toda a carga fosse desembarcada (Fitzgerald, 1986).

Com a introdução dos contêineres, foi possível carregar e descarregar mercadorias em poucas horas, sem tocá-las individualmente Fitzgerald (1986). A padronização permitiu com que os contêineres fossem eficientemente empilhados tanto em navios, quanto em trens, caminhões e guindastes em todo o mundo.

A adoção dos contêineres, segundo pesquisa feita por Bernhofen et al. (2013), explica um aumento de 320% no comércio bilateral nos países industrializados passados cinco anos de sua adoção, e um aumento de 790% passados 20 anos.

Dois gráficos extraídos de (D. Steenken, 2004), dados nas figuras 3 e 4, ajudam a reforçar o crescimento do transporte de carga através de contêineres nos últimos anos.

Da Figura 3 é possível destacar que dos dez maiores portos seis estão localizados na Ásia, três na Europa Ocidental e um nos Estados Unidos. Já a Figura 4 reforça a importância de se considerar o transporte via contêineres para maior inserção no comércio internacional.

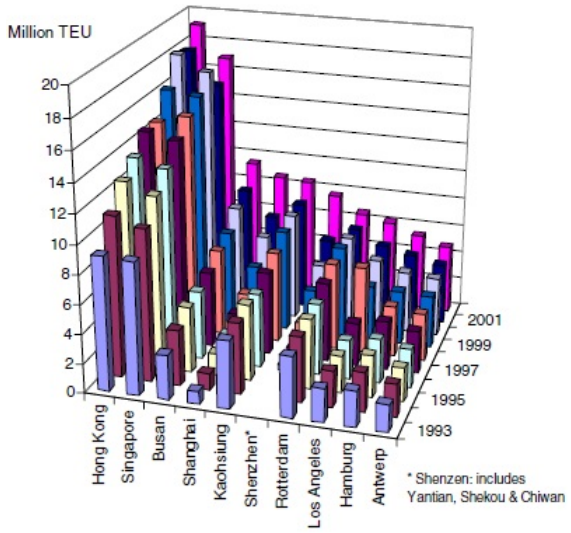


Figura 3: Número de contêineres movimentados nos dez maiores portos do mundo de 1993 até 2000 (Ranking de 2002). Fonte: D. Steenken (2004).

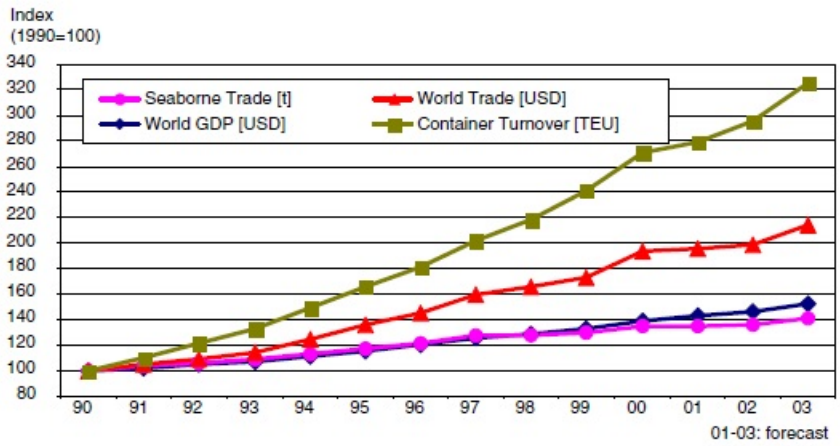


Figura 4: Evolução do transporte de cargas no mundo. Fonte: D. Steenken (2004).

Apesar, do sistema portuário brasileiro ter ao todo 34 portos, espalhados por cerca de 7.400 km de costa marítima, estes possuem infraestrutura precária e estrutura de fiscalização considerada por especialistas burocrática e obsoleta (ANTAQ, 2013). A Figura 5 ajuda a dar uma perspectiva da localização dos 34 principais portos do Brasil.

A ineficiência dos portos prejudica a participação do Brasil nos mercados mundiais. Ela é resultado de diversos fatores, dentre eles o modelo administrativo impróprio, atrasos derivados de altas concentrações de tráfego nas rodovias

Figura 5: Os 34 principais portos do Brasil. Fonte: ANTAQ (2013).

de acesso, equipamentos obsoletos e finalmente, o de maior importância, a falta de investimentos em infraestrutura.

Segundo pesquisa realizada pelo Bank (2008), o porto brasileiro mais importante é o porto de Santos, não apenas especificamente pela sua capacidade de carregamento, mas também em virtude da sua influência sobre a economia nacional. As particularidades de localização estratégica e da capacidade total de carga utilizadas, resultam na movimentação de aproximadamente 6,5% do PIB do país neste porto.

Segundo reportagem da Revista Exame (2011), os países do BRICS, por exemplo, apresentam índices de investimento muito maiores que os do Brasil (Vettorazzo, 2011). A China investe 9% do PIB em infraestrutura de transporte de carga, Índia e Rússia 5%, já o Brasil investiu somente $0,8\%$ nos últimos 10 anos (Vettorazzo, 2011). Em 2013 foi inaugurado na China um terminal capaz de movimentar 30 milhões de contêineres, equivalente a mais de 10 vezes a o que o porto de Santos conseguiu movimentar em 2010 ($2,8$ milhões).

Apenas 11% das cargas brasileiras são transportadas por navios, enquanto na China é 48%, na Europa 37% e no EUA 28%. Em 1999 o custo de movimentação de um container no Brasil era cerca de 500 dólares em 2009 este custo baixou para aproximadamente 200 dólares. Em Shangai na China, o custo de movimentação é em média de 75 dólares, em Roterdã 110 dólares, e a média de Santos é de 290 dólares (Maia, 2013).

Neste cenário, urge lançar mão de estudos para melhorar a efiência dos portos brasileiros. Este estudo tem um objetivo mais simples, mas que também pode auxiliar na realidade brasileira atual. O objetivo deste estudo é fornecer algumas contribuições de modelos matemáticos e técnicas de solução com vistas a incrementar a capacidade operativa de portos de contêineres.

2 Modelos Matemáticos para Terminais Portuários de Contêineres

Segundo Murty et al. (2005), um terminal contêiner serve como uma interface entre o oceano e o transporte terrestre e tem duas funções principais: receber contêineres de exportação para embarcar nos navios e descarregar contêineres de importação de embarcações para entregar aos consignatários. Logo, sua eficiência é essencial para permitir o incremento do fluxo de contêineres em uma cadeia global de suprimentos.

De acordo com Guan et al. (2013); D. Steenken (2004); R. Stahlbock (2008), as operações de um terminal de contêineres podem ser divididas em cinco problemas principais:

1. Alocação de Berços: Programação da atribuição de berços no cais do porto para permitir o atendimento de navios;
2. Plano de Estiva: Programação de Carregamento e descarregamento de contêineres entre um navio e um berço do porto, por meio de um ou mais guindastes portuários (*Quay Cranes*), observando um certo número de portos a serem percorridos;
3. Atribuição e *Scheduling* de Guindastes Portuários: Atribuição de guindastes portuários (*Quay Cranes*) de modo a se elaborar uma programação de operação para cada seção do navio;
4. Transporte no Cais: Transporte dos contêineres de cada berço para o pátio do porto por meio de máquinas apropriadas tais como *Automated Guided Vehicles (AGV)* e caminhões;
5. Transporte no Pátio: Carregamento e descarregamento de contêineres no pátio do porto (embarque nos navios ou o despacho para caminhões e trens) por meio de equipamentos como guindastes de pátio (*Yard Cranes*), pórticos (*Straddle Carriers*), e alguns tipos de empilhadeiras: *forklift* ou *reach stackers*).

Uma visão geral sobre o porto que engloba todos os cinco problemas destacados anteriormente é fornecida pela Figura 6.

Para o problema (1) é esperado uma saída de informações como a dada na Figura 7.

Na Figura 7 é possível verificar que o agendamento de navios para cada berço no tempo deve também contemplar aspectos referentes à distância de segurança (caso dos navios 6 e 5) e disponibilidade dos berços (caso dos navios 1, 3 e 4). Existem outros aspectos que devem ser levados em consideração no tempo de atracação do navio em um berço. Um primeiro aspecto consiste em considerar todo o maquinário necessário para carregar ou descarregar os contêineres de um navio para o pátio porto ou vice-versa. Neste caso, devem ser abarcados os problemas (3) e (4) cujos elementos são descritos na Figura 6.

Um segundo aspecto que deve ser considerado é problema (2), o plano de estiva. A elaboração do plano de estiva está relacionada com a estrutura celular que o navio porta-contêineres possui tal como dado na Figura 8.

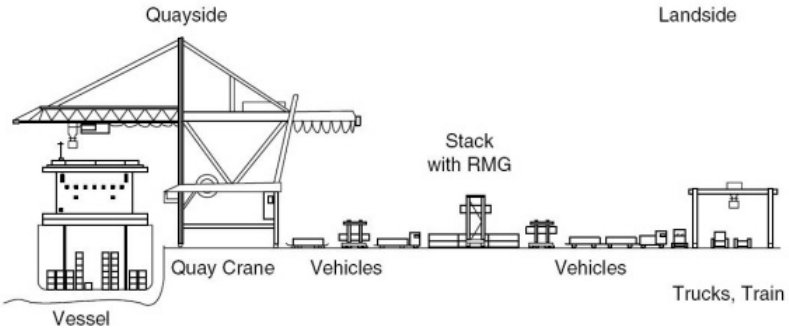


Figura 6: Representação esquemática dos elementos de um terminal portuário de contêineres. Fonte: D. Steenken (2004).

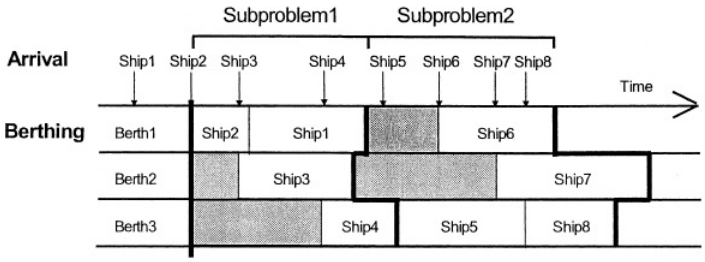


Figura 7: Programação da atribuição de 3 berços para atender 8 navios. Fonte: Nishimura et al. (2001).

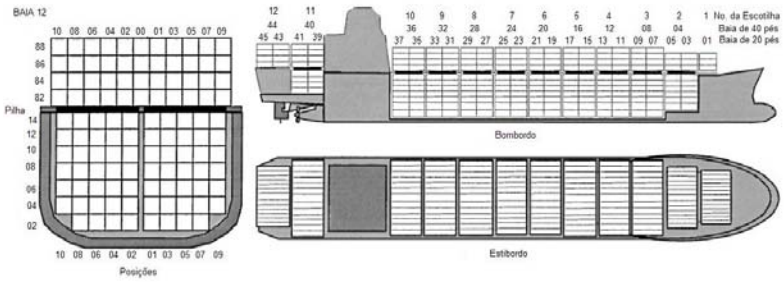


Figura 8: Estrutura Celular de um navio. Fonte: Wilson & Roach (2000).

A estrutura celular do navio é tal que só é possível acessar um contêiner por meio de uma pilha específica. Assim, pode ocorrer ao se mover um contêiner, outros devem ser movidos para permitir a retirada do contêiner que deve ficar no porto atual, mas deverão ser devolvidos ao navio. O movimento de retirada e devolução de um contêiner para um navio é dito remanejamento. Se o remanejamento ocorrer com frequência, então, isso irá implicar em um maior tempo de atracação do navio no berço. Para evitar tal transtorno é necessário

elaborar o plano de estiva de modo que a decisão um porto não acarrete muitos movimentos de remanejamento nos próximos portos a serem visitados.

Para realizar o carregamento ou descarregamento de contêineres em navios é necessário empregar guindastes portuários (*quay cranes*). A alocação destes equipamentos deve obedecer uma série de requisitos operativos como, por exemplo, um distância mínima de segurança entre dois guindastes. Assim, a programação que define, para um certo horizonte, a operação de um equipamento pode sofrer interferência da condição ou mesmo operação de outros guindastes. A Figura 9 fornece um indicativo das três informações relevantes neste problema: o número e a posição dos guindastes em cada baia do navio, e o trabalho a ser realizado em cada baia.

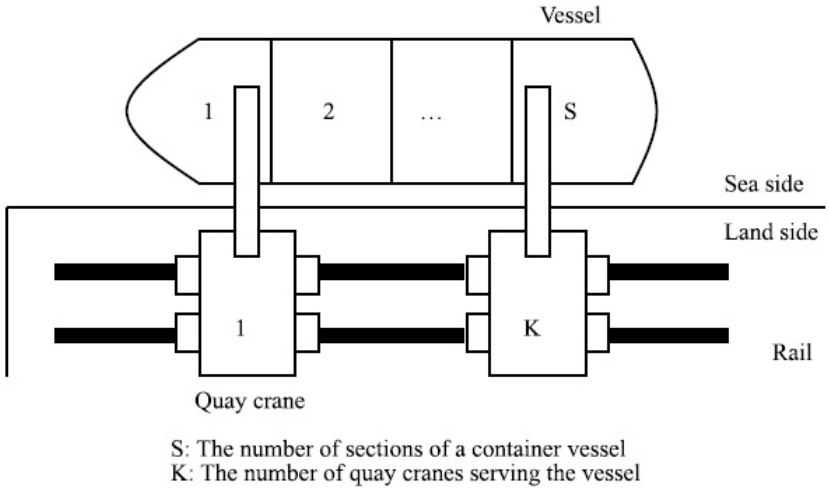


Figura 9: Disposição de contêineres em um pátio portuário (Fonte: (Guan et al., 2013)).

Após executar o plano de estiva por meio da alocação de guindastes portuários, é necessário, então, realizar a movimentação da carga pelo pátio portuário. Vários são os equipamentos e as decisões a serem tomadas. Para ilustrar a complexidade associada a movimentação de cargas no pátios, os elementos dos problemas (4) e (5) são detalhados na Figura 10. Além disso, o artigo (Carlo et al., 2014) é indicado como uma boa revisão acerca do tema, pois fornece, também, um esquema de classificação sobre 91 trabalhos, publicados entre 2004 e 2012, na área de movimentação de cargas em pátios portuários de acordo com o seguintes critérios: quais são as variáveis de decisão, qual o tipo de *layout* do pátio, como é realizada o despacho e roteamento dos equipamentos, informação sobre a chegada e a saída de contêineres, uso ou não de otimização estocástica, e medidas de desempenho empregadas como função objetivo. É importante destacar que neste artigo são citadas nove áreas promissoras para pesquisas futuras e dentre elas se destacam duas: a identificação rápida se o pátio é ou não um gargalo e a integração entre a alocação de espaço no pátio com a operação de guindastes portuários.

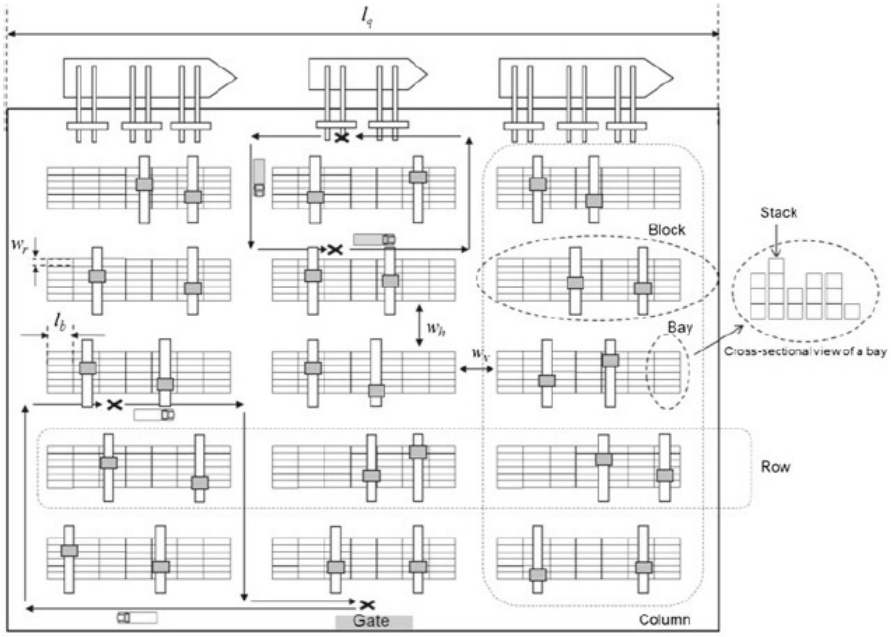


Figura 10: Disposição de contêineres em um pátio portuário (Fonte: (Lee & Kim, 2013)).

A consideração do problema (5) é importante para fornecer uma melhor estimativa do tempo necessário de atracação do navio para realizar as operações de acordo com o plano de estiva obtido ao se resolver (2).

Para cada um dos cinco problemas existe um quantidade considerável de metodologias e modelos como pode ser verificado em (R. Stahlbock, 2008). O texto a seguir irá cobrir alguns destes cinco problemas. Na seção 3 será realizada a descrição e a resolução somente do problema de alocação de berços (problema (1)). A seção 4 trata apenas da formulação e uma proposta de resolução do problema de estiva (problema (2)). A seção 5 irá considerar a integração do problema de estiva com o problema de alocação de guindastes portuários (problemas (2) e (3)).

3 O problema de alocação de berços

Em muitos portos é importante estabelecer qual ordem e como os navios serão descarregados ou carregados.

O problema de alocação de navios em berços consiste em alocar os navios ao longo dos berços do porto, e, possui duas principais decisões : *onde* e *quando* os navios devem atracar.

Normalmente, existem restrições de atracação (devido a profundidade de cada berço), também com relação a distância máxima entre guindaste e navio, ou até mesmo se o berço é especializado em um tipo de navio específico.

Essa escolha visa minimizar o tempo de espera dos navios no porto.

Definindo:

- N: set of vessels, $n = |N|$;
- M: set of berths, $m = |M|$;
- t_i^k: duração do atendimento do navio i no berço k;
- a_i: horário de chegada do navio i;
- s^k: horário de abertura do berço k;
- e^k: horário de fechamento do berço k;
- b_i: horário de termino da janela de tempo para o navio i;
- v_i: valor do tempo de serviço do navio i;
- $x_{ij}^k \in \{0,1\} \forall k \in M, \forall (i,j) \in A^k, x_{ij}^k = 1$ se o navio j é atendido pelo berço k após o navio i;
- $x_{ij}^k \in \{0,1\} \forall k \in M, \forall (i,j) \in A^k, x_{ij}^k = 1$ se o navio j é atendido pelo berço k após o navio i;
- $T_i^k \forall k \in M, i \in N$ é o horário que o navio i atracou no berço k;
- $T_{o(k)}^k \forall k \in M$ é o horário em que o primeiro navio atracou no berço k;
- $T_{d(k)}^k \forall k \in M$ é o horário em que o último navio saiu do berço k;
- $M_{ij}^k = \max\{b_i + t_i^k - a_i\}, \forall k \in M, \forall (i,j) \in N$.

Segundo Cordeau et al. (2005) podemos descrever o problema de alocação de navios em berços matemáticamente da seguinte forma:

- a função objetivo (1a) minimiza a soma dos intervalos de tempos;
- a restrição (1b) garante que cada navio é atendido por apenas um berço;
- as restrições (1c) e (1d) garantem, respectivamente, que um navio será o primeiro a ser atendido em cada berço e outro será o último;
- a restrição (1) garante a "conservação do fluxo", ou seja, o atendimento para os demais navios;

- a restrição (1) faz o cálculo do horário de atracação dos navios, sendo considerados apenas os navios válidos para cada berço, ou seja, alguns navios não podem ser atendidos em determinados berços devido a restrições técnicas;
- as restrições (1g) e (1) garantem, respectivamente, que o horário de atracação seja após a chegada do navio e que o horário do término do atendimento do navio seja anterior ao horário-limite do navio (janela de tempo);
- as restrições (1i) e (1j) garantem a não violação das janelas de tempo nos berços;
- por fim, a restrição (1k) garante que as variáveis de decisão sejam binárias.

$$\min \sum_{i \in N} \sum_{k \in M} v_i \left[T_i^k - a_i + t_i^k \sum_{j \in N \cup \{d(k)\}} x_{i,j}^k \right] \tag{1a}$$

sujeito a

$$\sum_{k \in M} \sum_{j \in N \cup \{d(k)\}} x_{ij}^k = 1 \qquad \forall i \in N \tag{1b}$$

$$\sum_{j \in N \cup \{d(k)\}} x_{o(k)j}^k = 1 \qquad \forall k \in M \tag{1c}$$

$$\sum_{i \in N \cup \{o(k)\}} x_{i,d(k)}^k = 1 \qquad \forall k \in M \tag{1d}$$

$$\sum_{j \in N \cup \{u(k)\}} x_{i,j}^k - \sum_{j \in N \cup \{o(k)\}} x_{j,i}^k = 0 \tag{1e}$$

$$\forall k \in M, \forall i \in N$$

$$T_i^k + t_i^k - T_j^k \leq (1 - x_{i,j}^k) M_{i,j}^k \tag{1f}$$

$$\forall k \in M, \forall (i,j) \in A^k$$

$$T^K \geq a_i \qquad \forall k \in M, \forall i \in N \tag{1g}$$

$$T_i^k + t_i^k \sum_{j \in N \cup \{d(k)\}} x_{j,i}^k \leq b_i \tag{1h}$$

$$\forall k \in M, \forall i \in N$$

$$T_{o(k)}^k \geq s^k \qquad \forall k \in M \tag{1i}$$

$$T_{d(k)}^k \leq e^k \qquad \forall k \in M \tag{1j}$$

$$x_{i,j}^k \in \{0,1\} \qquad \forall k \in M, \forall (i,j) \in A^k \tag{1k}$$

Essa função objetivo minimiza a soma dos tempos de atendimentos de acordo com um peso v_i.

3.1 Abordagem via Algoritmo Genético com chaves aleatórias viciadas e *Clustering Search*

Nesse trabalho o Problema de Alocação de Berços (PAB) é tratado de forma semelhante ao apresentado por Cordeau et al. (2005), isto é, em sua forma discreta e dinâmica, onde o cais é dividido em um conjunto finito de berços e os navios podem chegar a qualquer momento. Esse modelo foi escolhido, por um lado, considerando-se que a maioria dos principais portos brasileiros opera através de berços e, por outro, devido à natureza dinâmica dos mesmos, pois as condições e restrições dos berços não são imutáveis. Desse modo, os tempos descritos na Figura 11 representam as condições reais vividas nos portos: quando um navio i chega ao porto, ele deve esperar até que seja autorizada sua atracação num berço k. A diferença entre o horário de chegada (a_i) e o de atracação (T_i^k) é chamado de tempo de espera. Durante o carregamento e descarregamento de carga é contabilizado o tempo de atendimento (t_i^k) que pode ser obtido pela diferença entre o horário de saída e o de atracação. O período total desde a chegada do navio até sua saída é chamado de tempo de serviço.

Figura 11: Cronograma de atendimento de um navio. Fonte: Cordeau et al. (2005)

Neste estudo, buscou-se uma alternativa que pudesse atender o PAB de forma satisfatória, ao mesmo tempo em que fosse o mais independente possível do mesmo; justamente para poder ser aplicada a outros problemas de otimização com um mínimo de esforço de adaptação. Assim, propõe-se o BRKGA+CS, um algoritmo híbrido baseado nas meta-heurísticas Clustering Search (CS) Oliveira et al. (2013) e Biased Random Key Genetic Algorithm (BRKGA) Gonçalves & Resende (2011). O CS se mostra eficiente na busca de regiões promissoras do espaço de soluções e na intensificação através de métodos de otimização local e, o BRKGA apresenta características genéricas, adaptando-se facilmente ao problema em análise, pois a maioria de seus componentes independe do mesmo. Desta forma, busca-se manter a robustez e eficiência do CS, simplificando, no entanto, sua implementação.

3.1.1 BRKGA

O BRKGA é uma meta-heurística recente e baseada no Algoritmo Genético de Chaves Aleatórias (*Random Keys Genetic Algorithm* (RKGA), em inglês) proposto por Bean (1994) para resolver problemas de sequenciamento, tais como

"*multiple machine scheduling problems*" e "*resource allocation problems*". Entretanto, conforme observado por Amorim (2011), seu bom desempenho motivou a comunidade científica a testá-lo numa grande variedade de problemas combinatórios, onde as soluções podem ser representadas como vetores de permutação. O RKGA é formado por indivíduos cujos genes são números reais gerados aleatoriamente dentro do intervalo $]0, 1]$. Esses números são chamados de Chaves Aleatórias ("Random Keys", em inglês). A codificação de um cromossomo é realizada pela escolha aleatória de n valores reais entre $]0, 1]$, sendo n o número de alelos do cromossomo e que está relacionado intrinsecamente ao problema em análise. É importante ressaltar que esse vetor de chaves aleatórias forma um cromossomo para o RKGA e não uma solução para o problema.

Assim, para se realizar um mapeamento entre o espaço das chaves aleatórias e o espaço real do problema, faz-se o uso de um algoritmo determinístico, chamado de decodificador ("decoder", em inglês) que é responsável em retornar uma solução viável e seu respectivo valor objetivo ou de aptidão ("fitness") a partir de um vetor de chaves aleatórias. Por essa razão, cada problema de otimização tem um decodificador específico.

O RKGA pode ser descrito, de forma mais detalhada, como um conjunto de p vetores constituídos de n chaves aleatórias, os quais sofrem uma evolução ao longo de um número de iterações. Especificamente para a população inicial, cada componente de um vetor é gerado de forma independente e ao acaso (Buriol et al., 2010). Na sequência, o decodificador atua em cada indivíduo da geração k calculando sua aptidão ("fitness"). Com base nessas referências a população é dividida em dois grupos: um pequeno grupo, chamado "elite", formado pelos pe indivíduos com os melhores valores de "fitness" e, o outro grupo com o restante da população, composto dos $p - pe$ indivíduos.

No processo de evolução, copiam-se todos os indivíduos "elite" da população da geração atual (k) para a população da próxima geração, $(k + 1)$. Outra parcela dessa nova população é formada por pm indivíduos mutantes que garantem diversidade e assim auxiliando o algoritmo a escapar de mínimos locais (Gonçalves & Resende, 2011). Um mutante é formado da mesma maneira que um elemento da população inicial. Por fim, para completar a nova população, são realizados cruzamentos através da combinação de pares de indivíduos da população atual; de modo a produzir $p - pe - pm$ indivíduos (Buriol et al., 2010).

A diferença marcante entre RKGA e o BRKGA está exatamente no método de seleção de indivíduos que participarão do cruzamento e que foi detalhado por Gonçalves & Resende (2011): enquanto que no RKGA dois pais são selecionados, de forma aleatória do total da população, no BRKGA, necessariamente, o primeiro (pai A) é escolhido aleatoriamente dentre os indivíduos do grupo da elite e o outro (pai B) é escolhido do grupo não elite. Isto é, a escolha é aleatória e tendenciosa (Amorim, 2011).

Nos dois métodos, a combinação dos pais é feita com o método uniforme parametrizado onde o filho herda a i-ésima chave do pai A com probabilidade $R(E) > 0,5$ e a do pai B com probabilidade $1 - R(E)$ (Bean, 1994; Gonçalves & Resende, 2011). Numa análise mais detalhada, é possível verificar que o BRKGA possui duas partes distintas: uma consistindo do algoritmo genético com seus métodos de tratamento dos cromossomos e gerações, denominada problema independente e outra consistindo do decodificador ("decoder"), denominada problema dependente, conforme estrutura básica da Figura 12.

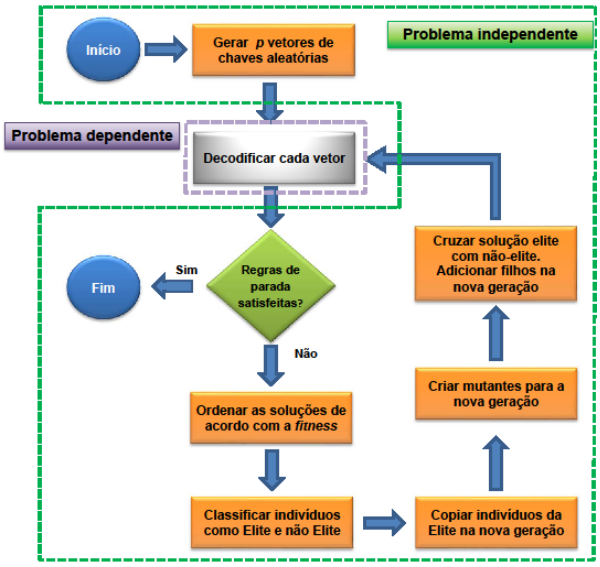


Figura 12: Fluxograma do BRKGA. Fonte: Gonçalves & Resende (2011).

3.1.2 CS

As meta-heurísticas utilizadas para resolver problemas de otimização combinatória muitas vezes não conseguem alcançar os resultados esperados num tempo computacional viável, principalmente, para uma aplicação prática. Por outro lado, a ideia de aplicar uma heurística de busca local específica nas soluções das meta-heurísticas visando melhorar seu desempenho, poderia gerar o efeito contrário, tornando o processo de busca impraticável em relação ao tempo de processamento. Oliveira et al. (2013), motivado por essas considerações, propõe o método híbrido CS que, munindo do conceito de regiões promissoras, procura racionalizar a aplicação dos processos de intensificação de busca, sendo somente aplicados após identificar regiões promissoras (com grande potencial de melhoria das soluções).

O CS possui três componentes principais: um processo de agrupamento, uma meta-heurística geradora de soluções e, uma heurística de otimização local (Oliveira et al., 2013).

A função do processo de agrupamento é dividir o espaço de busca em regiões e, através da classificação de padrões, agrupar as soluções com características semelhantes. De acordo com Costa & de Oliveira (2013), cada grupo, ou cluster é definido por três atributos: o centro c_j que identifica a localização do cluster j dentro do espaço de busca e é representado por uma solução que possui características das soluções similares até então geradas pela meta-heurística; o volume v_j que monitora a qualidade do cluster j, que terão seus volumes aumentados no decorrer das iterações do algoritmo, entretanto, aqueles mais promissores o farão de forma mais rápida. Assim, um cluster se torna interessante quando seu volume atinge um limitante λ, definido a priori e, por fim, o índice de ineficácia r_j responsável por evitar que a busca local fique sendo executada em regiões ruins ou que já tenham sido suficientemente exploradas.

Nesse processo, a cada iteração, é preciso identificar semelhanças entre uma solução gerada pela meta-heurística (s_k) e os centros dos clusters. Para tanto, define-se uma função de medida de distância $d(i, j)$ que retorna um número positivo. Esse valor representa, de forma inversamente proporcional, a similaridade entre duas soluções. Isto é, quanto maior o valor de $d(i, j)$ menor a similaridade entre as soluções e vice-versa (Chaves, 2009).

A função da meta-heurística é gerar, a cada iteração, soluções que são enviadas ao processo de agrupamento. Chaves (2009) ressalta que ela deve ser capaz de gerar um grande número de soluções diferentes, considerando principalmente os critérios de diversificação e velocidade de produção de soluções. Assim, se garante uma investigação mais abrangente do espaço de busca.

Quando um cluster assume a condição de "promissor" uma heurística de otimização local realiza uma intensificação da busca no seu centro.

A Figura 13 traz um fluxograma do CS contendo as etapas principais. Destacam-se os processos de agrupamento e de otimização local. Conforme indicado, os clusters iniciais precisam ser criados antecipadamente. Esse procedimento consiste em gerar aleatoriamente ou através de uma heurística, uma solução viável que representará a localização de um cluster no espaço de busca. Chaves (2009) ressalta a importância da quantidade de clusters (NC) que serão criados, pois um pequeno número pode ocasionar uma análise restrita a poucas regiões do espaço de busca, por outro lado, um número elevado poderá tornar o processo inviável computacionalmente.

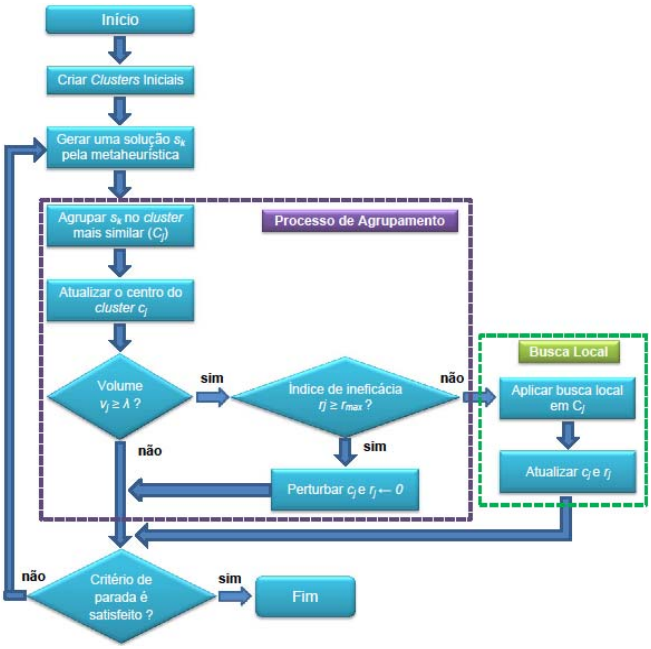


Figura 13: Fluxograma do método CS. Fonte: Chaves (2009).

Na sequência, uma solução s_k gerada pela meta-heurística é enviada ao processo de agrupamento. Após definido o cluster j mais similar, essa solução é

incorporada ao mesmo e o respectivo centro c_j é atualizado com as características da solução s_k, através do processo de assimilação. Nesse processo, o centro c_j sofre um deslocamento no espaço de busca. O próximo passo é incrementar o volume v_j em uma unidade e compará-lo ao limitante λ. Caso seja inferior, o cluster j não pode ainda ser considerado promissor, caso contrário, verifica-se se a variável de controle r_j desse cluster atingiu o limitante de ineficácia rmax. Em positivo, aplica-se uma perturbação aleatória no centro c_j com o intuito de afastá-lo dessa região do espaço de busca e a variável r_j é reiniciada. Caso contrário, intensifica-se a busca na vizinhança desse cluster através do algoritmo de otimização local. Não obtendo êxito, a variável r_j é acrescida em uma unidade. Findo o processo de agrupamento, retorna-se à meta-heurística. Recentemente, Oliveira et al. (2013) simplificaram o método retirando o índice de ineficácia r_j, assim toda vez que o volume atinge o limitante λ o componente de otimização local é aplicado.

3.1.3 BRKGA+CS

O emprego do BRKGA como meta-heurística geradora de soluções para o CS tem a intenção de generalizar o processo, tornado a implementação do CS mais simples. Isto é, as soluções fornecidas ao CS não são decodificadas antecipadamente, e sim enviadas no formato de um vetor de chaves aleatórias para o componente de agrupamento do mesmo. Dessa forma, com exceção das buscas locais, todos os outros componentes do CS trabalharam nesse formato, ou seja, independente do problema em análise. Somente no final, após decodificação, é retornada uma solução para o problema. Conforme ilustra a Figura 14.

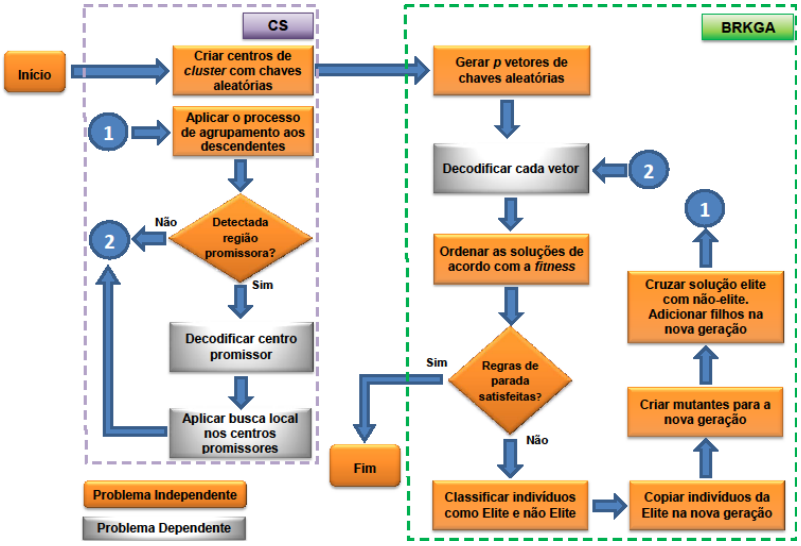


Figura 14: Fluxograma do método BRKGA+CS.

Neste trabalho, a decodificação das chaves aleatórias baseou-se na quantidade de berços (b) e navios (n) das instâncias do PAB analisadas. Assim, o intervalo]0,1] é dividido igualmente em b faixas. Por exemplo, para $b = 2$, a

faixa $]0; 0,5]$ representaria o Berço 1 enquanto que $]0,5; 1]$, o Berço 2. Para $n = 5$, um possível vetor seria $\{0,06; 0,98; 0,93; 0,85; 0,16\}$ e que, após ordenação, resultaria na seguinte sequência de atracação: Berço 1: navios 1 e 5; Berço 2: navios 4, 3 e 2.

O processo de assimilação ficou a cargo do método Reconexão por Caminhos (PR, do inglês *Path-Relinking*) (Glover & Martí, 2000). Nele são utilizadas duas soluções conhecidas. A primeira, nomeada "inicial", é de onde parte o procedimento exploratório. A segunda, chamada "guia", fornece características ou atributos que serão utilizados para gerar as soluções intermediárias. A intenção é encontrar uma solução intermediária (vizinha) melhor que as já conhecidas. Ressalta-se que, no caso do CS, o centro será deslocado para a melhor solução avaliada nessa trajetória mesmo que ela seja pior que o centro corrente.

A medida de similaridade, que traduz o grau de semelhança entre as soluções, foi a distância euclidiana, cujo valor (de) é calculado diretamente nas chaves aleatórias.

$$de = \sqrt{\sum_{i=1}^{n} \left(ch2_i - ch1_i\right)^2}$$

sendo, n o número de alelos do cromossomo (tamanho do vetor), $ch2_i$ o valor da i-ésima chave do vetor da solução 2 e $ch1_i$ o valor da i-ésima chave do vetor da solução 1.

Ressalta-se o fato desses componentes atuarem diretamente nos vetores de chaves aleatórias, reforçando o benefício do BRKGA na simplificação do CS.

As penalizações utilizadas foram as mesmas apresentadas por Mauri et al. (2008), ou seja: $[\omega_0, \omega_1, \omega_2] = [1, 10, 10]$, assim como as buscas locais definidas pelos movimentos Reorde-nar, Realocar e Trocar. Maiores detalhes sobre esses movimentos são apresentados pelos autores.

Inicialmente foi realizado um processo de calibragem dos parâmetros visando obter o melhor desempenho dos métodos. Desta forma, os parâmetros utilizados nos testes computacionais foram os que obtiveram os melhores resultados para o BRKGA e o BRKGA+CS. Assim para o BRKGA foram: $p = 100$; $pe = 0,20$; $pm = 0,20$ e $\rho_e = 0,65$. E, para o BRKGA+CS: $p = 200$; $pe = 0,25$; $pm = 0,15$ e $\rho_e = 0,65$. Os do CS foram: $NC = 20$, $\lambda = 4$ e $rmax = 300$.

A estratégia de refinamento das soluções no processo de otimização local, foi a Variable Neighborhood Descent (VND) (Mladenovic & Hansen, 1997) onde as heurísticas citadas anteriormente são executadas conforme descrito, de forma simplificada, pelo pseudocódigo da Figura 15. O centro do cluster promissor é decodificado em uma solução do PAB no inicio do processo de busca e após esse procedimento, a melhor solução encontra é decodificada novamente no centro do cluster.

3.1.4 Experimentos computacionais

Inicialmente, a robustez e eficiência do BRKGA+CS podem ser comparadas com as do BRKGA através das curvas TTT, do inglês Time to Target, apresentadas na Figura 16. A intenção é verificar a contribuição do CS na obtenção de melhores resultados em um intervalo de tempo reduzido, tradução dos conceitos de intensificação local e regiões promissoras. As curvas cor-respondem ao tempo necessário para que os algoritmos alcançassem o valor alvo, que nesse caso foi $0,5\%$ acima do ótimo para a instância i02, ou seja, 1267. O algoritmo foi

```
1. DADO (S) FAÇA                    //Solução S vinda do CS
2. M ← 1;                           //Variável auxiliar
3. S* ← S;                          //Melhor solução
4. ENQUANTO (M ≤ 3) FAÇA
5.     CASO (M = 1) FAÇA;
6.           S' ← BL1(S*);          //Executa Busca Local 1 e armazena nova solução
7.     CASO (M = 2) FAÇA;
8.           S' ← BL2(S*);          //Executa Busca Local 2 e armazena nova solução
9.     CASO (M = 3) FAÇA;
10.          S' ← BL3(S*);          //Executa Busca Local 3 e armazena nova solução
11.    SE (f(S') < f(S*)) FAÇA;
12.        S* ← S';                 //Melhor solução até o momento
13.        M ← 1;                   //Retorna para a Busca Local 1
14.    SENÃO
15.        M ← M + 1;
16.    FIM-SE
17. FIM-ENQUANTO
```

Figura 15: Pseudocódigo do VND.

executado 100 vezes e as curvas são formadas pelos resultados encontrados na forma de probabilidade acumulada.

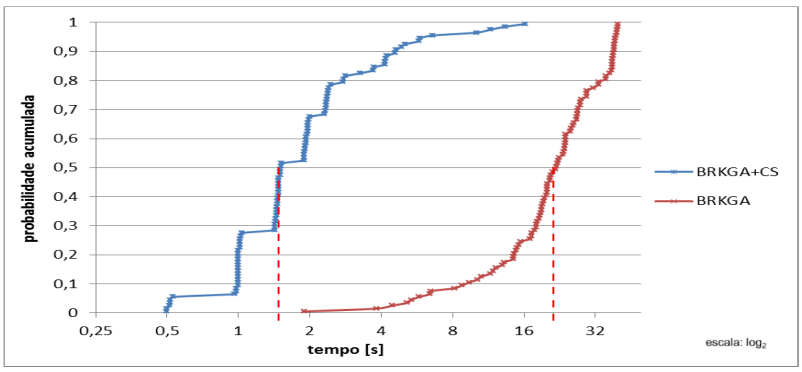


Figura 16: Probabilidade dos métodos atingir a solução alvo em determinado tempo de execução.

Observa-se que, na maioria das vezes, o BRKGA+CS encontrou o valor alvo antes de 2s, mais precisamente, 50% delas em até 1,5s. De forma análoga, percebe-se que o BRKGA demorou mais para encontrar o valor alvo, pois cerca de 50% das execuções levaram mais de 22s.

Nos experimentos computacionais realizados, assim como nas outras abordagens da literatura, foram utilizadas 30 instâncias, cada qual com 60 navios e 13 berços. Essas instâncias são baseadas em dados do porto de Gioia Tauro (Itália) e foram geradas randomicamente por Cordeau et al. (2005). Todos os testes computacionais foram executados em um PC com processador Intel Core i7-4770 com 3.40 GHz e 8 GB de memória RAM. Para cada instância foram realizados 30 testes aleatórios por método.

Na Tabela 1, o BRKGA+CS e o BRKGA são comparados a outras abordagens presentes na literatura. Nela estão relacionados os resultados obtidos por Cordeau et al. (2001) com uma heurística fundamentada na Busca Tabu (TS); Mauri et al. (2008) com um SA com reaquecimento (SA+RA), na sequência, o método GSPP proposto por Buhrkal et al. (2011) executado num PC Intel Xeon

de 2.66 GHz e por fim o CS de de Oliveira et al. (2012) que utiliza o SA como meta-heurística geradora de soluções. Os experimentos do CS e os do SA+RA foram realizados em um PC com processador AMD AthlonTM 64 de 2.2 GHz e 1GB de RAM. Cordeau et al. (2001) não especificaram a máquina utilizada em seus testes, e segundo o autor sua heurística utilizou, aproximadamente, 120 segundos para cada problema-teste.

As colunas (FO) e ([s]) contêm, respectivamente, o valor das soluções obtidas e os tempos de execução de cada algoritmo. Nas colunas "D1" e "D2" são listados os desvios entre as soluções do BRKGA+CS e BRKGA com relação às ótimas obtidas pelo CS+SA.

O BRKGA+CS, a exemplo do CS+SA e GSPP, alcançou os valores ótimos, porém com um tempo computacional superior. Mas, essa diferença se torna menos significativa em vista da característica genérica do BRKGA+CS, que pode ser adaptado facilmente para outros problemas de otimização. Assim como em Amorim (2011), o BRKGA sem o processo de intensificação de busca não alcançou os valores ótimos.

Tabela 1: Comparação dos resultados com outros métodos da literatura.

Instância	TS[1]		SA+RA[2]		GSPP[3]		CS+SA[4]		BRKGA+CS		BRKGA		Desvio [%]	
	FO	[s]	FO	[s]	FO	[s]	FO	[s]	FO	[s]	FO	[s]	D1	D2
I01	1415	-	1409	53,12	1409	17,92	1409	12,47	1409	20,94	1423	37,88	0,00	0,99
I02	1263	-	1261	58,94	1261	15,77	1261	12,59	1261	20,59	1264	37,38	0,00	0,24
I03	1139	-	1129	54,03	1129	13,54	1129	12,64	1129	20,63	1139	39,33	0,00	0,89
I04	1303	-	1302	67,33	1302	14,48	1302	12,59	1302	20,75	1309	39,17	0,00	0,54
I05	1208	-	1207	55,38	1207	17,21	1207	12,68	1207	20,39	1213	38,85	0,00	0,50
I06	1262	-	1261	53,88	1261	13,85	1261	12,56	1261	19,96	1264	38,38	0,00	0,24
I07	1279	-	1279	60,52	1279	14,60	1279	12,63	1279	20,25	1281	40,01	0,00	0,16
I08	1299	-	1299	61,45	1299	14,21	1299	12,57	1299	20,53	1311	38,49	0,00	0,92
I09	1444	-	1444	57,91	1444	16,51	1444	12,58	1444	20,62	1452	39,34	0,00	0,55
I10	1213	-	1213	68,95	1213	14,16	1213	12,61	1213	20,75	1224	40,53	0,00	0,91
I11	1378	-	1368	76,77	1368	14,13	1368	12,58	1368	20,43	1384	38,84	0,00	1,17
I12	1325	-	1325	62,84	1325	15,60	1325	12,56	1325	20,63	1357	38,77	0,00	2,42
I13	1360	-	1360	68,19	1360	13,87	1360	12,61	1360	20,48	1363	39,48	0,00	0,22
I14	1233	-	1233	75,06	1233	15,60	1233	12,67	1233	19,81	1236	41,40	0,00	0,24
I15	1295	-	1295	54,55	1295	13,52	1295	13,80	1295	20,58	1303	39,17	0,00	0,62
I16	1375	-	1364	63,91	1364	13,68	1364	14,46	1364	20,02	1383	38,95	0,00	1,39
I17	1283	-	1283	56,28	1283	13,37	1283	13,73	1283	20,29	1284	38,52	0,00	0,08
I18	1346	-	1345	53,98	1345	13,51	1345	12,72	1345	20,25	1355	38,53	0,00	0,74
I19	1370	-	1370	52,83	1367	14,59	1367	13,39	1367	20,70	1387	38,16	0,00	1,46
I20	1328	-	1328	53,38	1328	16,64	1328	12,82	1328	20,61	1346	38,33	0,00	1,36
I21	1346	-	1341	53,52	1341	13,37	1341	12,68	1341	21,06	1354	40,78	0,00	0,97
I22	1332	-	1326	57,97	1326	15,24	1326	12,62	1326	20,30	1354	37,36	0,00	2,11
I23	1266	-	1266	53,75	1266	13,65	1266	12,62	1266	20,99	1275	39,94	0,00	0,71
I24	1261	-	1260	54,09	1260	15,58	1260	12,64	1260	20,06	1262	38,00	0,00	0,16
I25	1379	-	1377	53,56	1376	15,80	1376	12,62	1376	21,09	1397	35,54	0,00	1,53
I26	1330	-	1318	57,34	1318	15,38	1318	12,62	1318	21,71	1331	35,30	0,00	0,99
I27	1261	-	1261	69,98	1261	15,52	1261	12,64	1261	20,27	1262	36,58	0,00	0,08
I28	1365	-	1360	58,47	1359	16,22	1359	12,71	1359	20,71	1380	38,41	0,00	1,55
I29	1282	-	1280	69,09	1280	15,30	1280	12,62	1280	20,59	1293	35,16	0,00	1,02
I30	1351	-	1344	70,67	1344	16,52	1344	12,58	1344	20,54	1366	34,99	0,00	1,64

[1] Cordeau et al. (2001) - [2] Mauri et al. (2008) - [3] Buhrkal et al. (2011) - [4] de Oliveira et al. (2012)

4 O problema do plano de estiva

A eficiência de um terminal portuário especializado em movimentação de contêineres depende da ordenação e agilidade do processo de lidar com os contêineres, especialmente durante o processo de carregamento dos navios. A estiva e o plano de carregamento associado são determinados fundamentalmente por dois critérios: instabilidade do navio e o número mínimo de remanejo requerido nos diversos pontos de entrega Ambrosino et al. (2006); Avriel et al. (2000); Dubrovsky et al. (2002); Wilson & Roach (2000). O último critério é baseado no fato de que muitos navios possuem uma estrutura celular, conforme pode ser observado na Figura 8, e os contêineres devem ser carregados de modo a formarem pilhas verticais, o que acarreta, em muitos casos, a necessidade de movimentar alguns contêineres para descarregar outros posicionados na parte inferior da pilha.

Concomitantemente, outra restrição emerge durante a escolha dos contêineres para carregamento no pátio do terminal, onde geralmente os contêineres são empilhados formando blocos a espera do momento de serem carregados. Se os contêineres alvos, que devem ser carregados mais tarde, são posicionados nas pilhas abaixo de outros, então a tarefa de carregamento requer remanejo de modo a remover e reposicionar os contêineres alvos. Esta situação ocorre com frequência, uma vez que a ordem de carregamento não é conhecida quando as cargas chegam ao pátio do terminal e é denominada de realocação. No entanto, mesmo quando esta informação é disponibilizada a tempo, o arranjo ideal de contêineres na área de armazenamento é praticamente impossível de ser obtido devido à chegada aleatória de diversas outras cargas. Mais especificamente, o problema de carregamento de contêineres em terminais portuários (PCCTP) consiste em determinar como carregar um conjunto de contêineres de diferentes tipos em um navio porta-contêiner (*containership*), respeitando restrições operacionais relacionadas aos contêineres, navio e pátio do terminal portuário.

Um navio porta contêiner tem sua capacidade medida em TEU (Twenty-foot Equivalent Units) ou Unidade Equivalente de Vinte pés. Por exemplo um navio com capacidade de 8000 TEUs pode carregar 8000 contêineres de vinte pés. Nos navios tem uma estrutura celular (vide Figura 8) onde são alojados os contêineres. Essas células são agrupadas por seções ou baias (em inglês bays) e os contêineres são empilhados nessas seções formando pilhas verticais. Então uma baia é um agrupamento de células, com capacidade de se empilhar um certo número de contêineres. A baia tem então linhas horizontais numeradas $r = 1, 2, ...R$, (a linha 1 é a linha que está em baixo, e a linha R é a linha do topo da pilha) e colunas numeradas $c = 1, 2, ..., C$ (coluna 1 é a primeira coluna da esquerda). A Figura 8 fornece uma da organização dos espaços no navio porta contêiner.

O PCCTP 3D que será resolvido aqui consiste em reduzir ao máximo possível dois objetivos: o número de realocações dos contêineres para um certo número de portos N e a instabilidade do arranjo dos contêineres. Para o primeiro objetivo, minimiza-se a realocação decorrente do descarregamento temporário de contêineres, da pilha de contêineres, com a finalidade de descarregar, num terminal portuário p, um contêiner que está na parte inferior da pilha. Isto é necessário porque os contêineres que estão numa pilha só podem ser acessados pelo topo. Então um contêiner que está no meio da pilha só pode ser descarregado num determinado porto p se os contêineres que estão acima dele forem removidos.

Para o segundo objetivo define-se a distância do centro de massa ao centro geométrico como medida da instabilidade do navio em cada porto Dubrovsky et al. (2002) . A seguir será apresentada uma formulação deste problema como sendo um problema de programação linear inteira com variáveis binárias $0-1$ que é uma extensão do modelo apresentado por Avriel et al. (1998). Maiores detalhes acerca da formulação proposta por Avriel et al. (1998) e a correspondente codificação em AMPL são fornecidos no Anexo A.1. Ambas as formulações respeitam as restrições operacionais relacionadas aos contêineres.

4.1 Modelo Matemático do Plano de Estiva

Considere um navio de transporte de contêineres que possui D baias numeradas $d = 1, ..., D$. Cada baia tem R linhas horizontais numeradas $r = 1, 2, ...R$, (a linha 1 é a linha que está em baixo, e a linha R é a linha do topo da pilha) e C colunas verticais numeradas $c = 1, 2, ..., C$ (coluna 1 é a primeira coluna da esquerda). Apesar da baia ter um formato tridimensional com baias de diferentes tamanhos, a mesma pode ser representada, sem perda de generalidade, por um formato tridimensional com baias de mesma capacidade, em particular um vetor de matrizes. Então, uma baia pode alocar no máximo $R \times C$ contêineres. É assumido também que todos os contêineres têm o mesmo tamanho e peso. O navio chega no porto 1 completamente vazio e sequencialmente ele visita os portos $2, 3, ..., N$. Em cada porto $i = 1, ..., N-1$, o navio recebe o carregamento de contêineres com destino aos portos $i+1, ..., N$. No último porto ele descarrega os contêineres e fica totalmente vazio. Seja $T = [T_{ij}]$ a matriz de transporte de dimensão $(N-1) \times (N-1)$, onde T_{ij} é o número de contêineres com origem em i e destino em j. A matriz é triangular superior porque $T_{ij} = 0$ para todo $i > j$.

A formulação de programação linear inteira do PCCTP 3D é dada pelas Eqs. (2)-(9).

$$\min f(x) = \alpha\phi_1(x) + \beta\phi_2(y) \tag{2}$$

subject to

$$\sum_{v=i+1}^{j}\sum_{r=1}^{R}\sum_{c=1}^{C}\sum_{d=1}^{D} x_{ijv}(r,c,d) - \sum_{k=1}^{i-1}\sum_{r=1}^{R}\sum_{c=1}^{C}\sum_{d=1}^{D} x_{kij}(r,c,d) = T_{ij}$$
$$i = 1, \cdots, N-1; j = i+1, \cdots, N \tag{3}$$
$$\sum_{k=1}^{i}\sum_{j=i+1}^{N}\sum_{v=i+1}^{j} x_{kjv}(r,c,d) = y_i(r,c,d)$$
$$i = 1, \cdots, N-1; r = 1, \cdots, R; c = 1, \cdots, C; d = 1, \cdots, D \tag{4}$$
$$y_i(r,c,d) - y_i(r+1,c,d) \geq 0$$
$$i = 1, \cdots, N-1; r = 1, \cdots, R; c = 1, \cdots, C; d = 1, \cdots, D \tag{5}$$
$$\sum_{i=1}^{j-1}\sum_{p=j}^{N} x_{ipj}(r,c,d) + \sum_{i=1}^{j-1}\sum_{p=j+1}^{N}\sum_{v=j+1}^{p} x_{ipv}(r+1,c,d) \leq 1 \tag{6}$$
$$x_{ijv}(r,c,d) = 0 ou 1$$
$$i = 1, \cdots, N; r = 1, \cdots, R; c = 1, \cdots, C; d = 1, \cdots, D. \tag{7}$$

onde: a variável binária $x_{ijv}(r,c,d)$ é definida de forma que assume o valor 1 se existir um contêiner no compartimento (r,c,d) que foi ocupado no porto i e tem como destino final o porto j e movido no porto v; caso contrário assume valor zero. Por compartimento (r,c,d) entende-se a linha r, a coluna c e a baia d no compartimento de carga do navio. É necessário salientar que a numeração das linhas é feita de cima para baixo, assim a linha de número 5 está abaixo da linha de número 4, a numeração das colunas é feita da esquerda para a direita e a profundidade é iniciada da proa até a popa. Similarmente, a variável $y_i(r,c,d)$ possui valor 1 se saindo do porto i o compartimento (r,c,d) for ocupado por um contêiner; caso contrário assume valor 0. A restrição (3) é a restrição de conservação de fluxo, onde T_{ij} é o elemento da matriz de transporte que representa o número de contêineres que embarcam no porto i com destino ao porto j. A restrição (4) em cada compartimento (r,c,d) será ocupado por no máximo um contêiner. A restrição (5) é necessária para garantir que existem contêineres embaixo do contêiner que ocupa o compartimento (r,c,d). A restrição (6) é responsável por definir a movimentação dos contêineres: se um contêiner que ocupa a posição (r,c,d) é descarregado no porto j, então, ou não existem contêineres acima dele, ou o índice v do contêiner que ocupa o compartimento $(r+1,c,d)$ não é maior que j.

A função objetivo da Eq. (2) possui é uma composição de dois diferentes critérios: o primeiro depende da movimentação dos contêineres, $\phi_1(x)$, e o segundo depende posição ocupada pelos contêineres em cada porto, $\phi_2(y)$. Os dois critérios podem ser combinados através de valores escalares fornecidos para os pesos α e β dentro de um modelo de otimização bi-objetivo. O termo $\phi_1(x)$ relativo ao custo total de movimentação dos contêineres (assumindo que a movimentação de um contêiner possui um custo unitário e igual para todos os portos) em todos os portos é dado pela Eq. (8).

$$\phi_1(x) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \sum_{v=i+1}^{j-1} \sum_{r=1}^{R} \sum_{c=1}^{C} \sum_{d=1}^{D} x_{ijv}(r,c,d) \tag{8}$$

O termo $\phi_2(y)$ é relativo à instabilidade do navio de acordo com a posição dos contêineres (assumindo que cada contêiner possui um peso unitário e igual para todos os portos) e calcula a distância entre o centro de massa e o centro geométrico de cada baia do navio depois de descarregar e depois de carregar em todos os portos tal como dado pela Eq. (9).

$$\phi_2(y) = \sum_{i=1}^{N} \left(\sum_{d=1}^{D} (\Delta x_{d,i})^2 + \sum_{d=1}^{D} (\Delta z_{d,i})^2 \right) \tag{9}$$

onde:

$$\Delta x_{d,i} = xm_{d,i} - R/2$$
$$\Delta z_{d,i} = zm_{d,i} - C/2$$
$$xm_{d,i} = \frac{\left(\sum_{r=1}^{R}\sum_{c=1}^{C} y_i(r,c,d)\cdot(r-0,5)\right)}{\left(\sum_{r=1}^{R}\sum_{c=1}^{C} y_i(r,c,d)\right)}$$
$$zm_{d,i} = \frac{\left(\sum_{c=1}^{C}\sum_{r=1}^{R} y_i(r,c,d)\cdot(c-0,5)\right)}{\left(\sum_{r=1}^{R}\sum_{c=1}^{C} y_i(r,c,d)\right)}$$

A formulação dada pelas Eqs ((2)-(9)) é um problema 3D e pode ser considerada uma extensão da formulação 2D apresentada por Avriel et al. (1998). Para a formulação 2D foi provado que é um problema NP-Completo Avriel et al. (2000) tal que o problema 3D também o é. A formulação 3D aqui apresentada é uma contribuição de Azevedo et al. (2014), tendo em vista que o modelo proposto Avriel et al. (1998) não considera a avaliação simultânea do número de movimento e da instabilidade através de uma função objetivo bi-objetivo.

Figura 17: Acidente no carregamento de contêines em um navio. Fonte: TCLN (2014).

A medida de instabilidade relativa ao arranjo de contêineres em um navio, pode ser, na verdade, mais complexa da que abordada pela função objetivo

Figura 18: Acidente no transporte de contêines com danos na estrutura do navio. Fonte: RNZN (2014).

anteriormente definida. Para além da questão do arranjo dos contêineres em cada baia com vistas a evitar acidentes, como o dado na Figura 17, é necessário considerar questões como estabilidade longitudinal, momento de inércia e estabilidade dinâmica para evitar problemas na estrutura do navio tais como o indicado na Figura 18. Ou ainda estabelecer uma distribuição dos contêineres que permitam a operação do navio em diversas condições climáticas sem que o mesmo aderne como dado na Figura 19. Neste sentido, é necessário empregar conceitos como centro de flutuação, meta-centro, dentre outros e a seguinte referência é indicada como introdução ao assunto (Derrett & Barrass, 1999).

Figura 19: Acidente no transporte de contêines com danos na estrutura do navio. Fonte: Oceans (2014).

A Figura 20 destaca alguns dos elementos que devem ser avaliados para se evitar que o navio seja colocado em uma posição de equilíbrio que leve o mesmo a adernar.

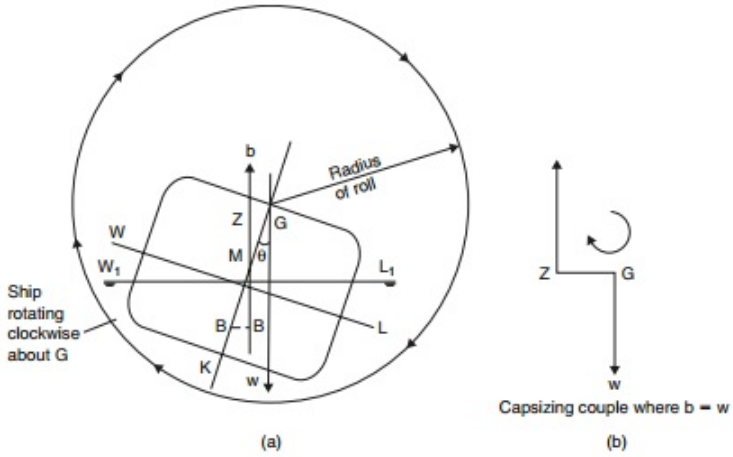


Figura 20: Análise de estabilidade. Fonte: Derrett & Barrass (1999).

De qualquer forma o assunto sobre estabilidade não é simples e em algumas situações a estabilidade pode ter de ser tratada como uma restrição "hard" e não simplesmente como uma restrição "soft", isto é, via função objetivo. Neste sentido, são indicados os trabalhos de Ambrosino et al. (2004, 2006); Delgado et al. (2012); Ning & Weijian (2009); D. Pacino (2012).

As mesmas desvantagens existentes na formulação 2D também ocorrem com a formulação 3D. Em particular, as formulações demandam um grande número de restrições e variáveis binárias para instâncias cujo porte é próximo de problemas reais. Para ilustrar essa afirmação, a representação de uma solução, através da formulação apresentada pelas Eqs ((2)-(9)), de uma instância que tenha $D = 5$, $R = 6$, $C = 50$, e $N = 30$ portos, demandará $D \times R \times C \times N^3$ variáveis $x_{ijv}(r, c, d)$, ou seja, 40.050.000 variáveis x, e $D \times R \times C \times N$ variáveis $y_i(r, c, d)$, ou seja, 45.000 variáveis y, somando um total de 40.545.000 variáveis para representar uma única solução. Deste modo, o problema assume um tamanho muito grande, tornando proibitivo o uso dessa formulação para problemas reais.

Uma alternativa para resolver este problema é empregar uma representação denominada de Representação por Regras, que reduz consideravelmente o número de variáveis. Para se ter uma ideia de quanto é esta redução, a instância que no exemplo acima necessita de 40.545.000 variáveis, na formulação apresentada, necessitará de 30 variáveis ao se utilizar à Representação por Regras, ou seja, o número de variáveis é igual ao número de portos. Além de ser compacta, a Representação por Regras tem assegura que todas as soluções geradas pelo método utilizado são factíveis. Ainda que a representação por regras seja empregada, o problema para encontrar a melhor combinação de regras também é um problema combinatório e é necessário empregar heurísticas para encontrar boas soluções também para o PCCTP 3D. Em particular, serão empregados o *Beam Search (BS)*, o Algoritmo Genético (AG) e o *Simulated Annealing (SA)*.

4.2 Representação por Regras

A principal motivação para empregar a representação por regras é observar o crescimento da quantidade de variáveis necessárias para se representar uma solução no modelo binário conforme cresce o número de portos. Isto ocorre porque o modelo binário apresenta uma descrição detalhada sobre como cada contêiner deve ser posicionado ou movido em cada porto para cada célula (r, c, d) do navio, mas sem garantir a consistência das várias restrições físicas:

- Ao longo dos portos, a quantidade de contêineres que entra no navio é igual a quantidade de contêineres que sai (Eq. (3)).
- Dois ou mais contêineres não podem ocupar o mesmo espaço (Eq. (4)).
- Contêineres não podem "flutuar" (Eq. (5)).
- A remoção de um contêiner em uma pilha só pode ser realizada se não houver outros contêineres impedindo sua retirada naquela pilha ou, se houver contêineres, que estes também sejam retirados (Eq. (6)).

Para evitar que soluções infactíveis sejam encontradas duas metodologias tradicionais são:

- Construir uma função de penalidade de modo a eliminar ou reduzir a quantidade de soluções infactíveis.
- Empregar uma estratégia de factibilização tal que soluções infactíveis sempre sejam transformadas em soluções factíveis.

A primeira metodologia encontra problemas na definição dos pesos para a penalização das soluções infactíveis, pois os valores podem ser dependentes para cada instância tratada. A segunda metodologia exige a elaboração de um mapeamento entre soluções infactíveis e factíveis que nem sempre é trivial e é dependente do problema.

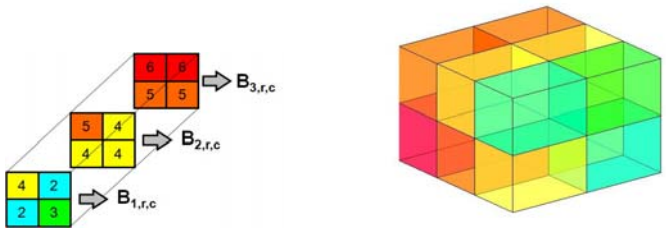
Uma alternativa é empregar uma representação que não se preocupa em detalhar a posição do contêiner e sim em uma descrição geral do que deve ocorrer em cada porto com duas das principais operações de um navio: o descarregamento (ao se chegar ao porto) e o carregamento (antes de sair do porto).

Para determinar os efeitos do que irá ocorrer com o arranjo do contêineres no navio, após uma regra de descarregamento ou uma regra de carregamento, será necessário empregar:

- Um vetor de matrizes B que representa o arranjo dos contêineres no navio em cada espaço (r, c, d). Essa matriz será modificada após a tomada de decisões em cada porto e representa uma memória de como as decisões tomadas em portos anteriores podem influenciar no arranjo do navio no porto atual.
- Um processo de simulação que transforma a instrução geral, ou regra, de carregamento ou descarregamento na movimentação de um contêiner específico e, portanto, modifica a matriz B. A partir do processo de simulação é possível extrair informações como o número de movimentos de

contêineres ou a medida de instabilidade do navio. O processo de simulação garante que a movimentação do contêiner seja realizada sem que haja violação das restrições físicas descritas anteriormente e, portanto, somente soluções factíveis são consideradas.

A justificativa para empregar o vetor de matrizes B está relacionada com a estrutura celular do navio (vide Figura 8), pois os locais onde os contêineres são alocados estão pré-determinados e devem formar pilhas verticais. Este empilhamento é o que permite usar o vetor de matrizes de ocupação B para fornecer a quantidade de espaços disponíveis e a localização dos contêineres no navio em cada porto. A matriz B_{drc} representa o estado de uma célula (r, c, d), isto é se $B_{drc} = 0$ significa que a célula que ocupa a baia d, a linha r e a coluna c está vazia e se $B_{drc} = j$ significa que a célula contém um contêiner cujo destino é o porto j. Assim, no Exemplo da Figura 21(a), o elemento $(1, 2, 2)$ pertence a baia 1 ($B_{1,r,c}$), linha 2 ($r = 2$) e coluna 2 ($c = 2$) é igual a 2 significando que neste local existe um contêiner que será descarregado no porto 2. De modo análogo, o elemento $B_{3,1,1} = 5$ significa que esta é uma célula da terceira baia ($B_{3,r,c}$), linha 1 ($r = 1$) e coluna 1 ($c = 1$) contém um contêiner cujo destino é o porto 5. Duas observações podem ser feitas: o vetor de matrizes B_{drc} mostrado na Figura 21(a) pode ser associado ao desenho tridimensional da Figura 21(b); a linha 1 representa o topo da pilha de carregamento e a linha 2 representa a parte inferior da pilha.



(a) Matriz B (b) Navio 3D

Figura 21: Como representar a matriz de ocupação B para o navio 3D.

É importante observar que a matriz de carregamento B é modificada em cada porto devido a entrada e saida de novos contêineres em cada porto, pois quando o navio chega num porto j é necessário realizar dois movimentos obrigatórios, a saber: descarregar os contêineres cujo destino é o porto j em questão e carregar os contêineres com destinos aos portos $j+1, j+2, ..., N$. Então, para cada porto j foram estabelecidas regras para se fazer o carregamento e descarregamento dos contêineres.

Muitas vezes, para se fazer o descarregamento no porto j, de um contêiner cujo destino é o porto j, é necessário fazer operações de remanejamento dos contêineres cujo destino são os portos de $j + 1$ até N, porque a posição que eles ocupam na pilha está acima da posição do contêiner do porto j. Veja por exemplo que na matriz B da Figura 21(b), para se descarregar os contêineres do porto 2, será necessário descarregar os contêineres contidos nas células $(1, 1, 1)$, $(1, 2, 1)$ e $(1, 2, 2)$. Assim, com intuito de reduzir o número de remanejamentos, ao se fazer o carregamento de contêineres num dado porto j deve-se levar em conta os contêineres que já estão no navio, porque foram embarcados nos portos

anteriores (portos de 1 até $j-1$) com destino aos portos $j+1$ até N. Observe então que existe uma relação íntima entre as operações de carregamento e descarregamento, tendo em vista que, a forma como é realizado o carregamento num porto j vai influenciar no descarregamento a ser efetuado nos demais portos (portos de $j+1$ até N). Portanto, para reduzir as operações de remanejamento é necessário estabelecer regras para o carregamento e descarregamento de contêineres em cada porto que leve em conta esta relação. Para tanto foram criadas doze regras, sendo seis para o carregamento (Rc1, Rc2, Rc3, Rc4, Rc5, Rc6) e duas para o descarregamento (Rd1, Rd2). A combinação de uma regra de carregamento com uma de descarregamento fornece a regra k para o porto j tal como dado na Tabela 2.

Tabela 2: Regras produzidas pela combinação de regras de carregamento e descarregamento.

Regras de Carregamento	Regras de Descarregamento	Regras
Rc1	Rd1	1
	Rd2	2
Rc2	Rd1	3
	Rd2	4
Rc3	Rd1	5
	Rd2	6
Rc4	Rd1	7
	Rd2	8
Rc5	Rd1	9
	Rd2	10
Rc6	Rd1	11
	Rd2	12

Observe na Tabela 2 que a regra 2 foi obtida utilizando a regra Rc1 para o carregamento dos contêineres e a regra Rd2 para o descarregamento. Isto foi feito com objetivo de se obter uma representação ainda mais compacta da solução.

A aplicação destas regras em cada porto j vai atualizar a Matriz de Ocupação B_{rcd} no porto j. Vale lembrar que inicialmente a matriz B está com todos seus elementos iguais a zero e ela começa a ser preenchida no porto 1. Para melhor ilustrar a utilização das regras, será utilizada a matriz de transporte T, que fornece a quantidade de contêineres que devem ser embarcados em cada porto i com destino a cada porto j, tal como dado na Figura 22. A capacidade e as dimensões adotadas para o navio são as mesmas apresentadas na Figura 21(a).

Para fins de clareza na descrição do funcionamento das regras será suposto que o navio está no porto 1 para as regras de carregamento e porto 2 para as regras de descarregamento.

	D2	D3	D4	D5
O1	2	5	0	0
O2	0	2	3	1
O3	0	0	2	2
O4	0	0	0	1

Figura 22: Matriz de transporte T.

4.2.1 Regras de Carregamento

- Regra Rc1: Esta regra preenche a matriz de ocupação B (no porto p) por baia, começando da última até a primeira linha, da esquerda para a direita, colocando na parte inferior da pilha de cada baia as cargas cujo destino é mais distante. A Aplicação desta regra considerando a matriz T da Figura 22 e que o navio se encontra no porto 1, resultará na matriz B da Figura 23(a).

- Regra Rc2: Esta regra preenche a matriz de ocupação B (no porto p) por linha começando da última até a primeira baia, da esquerda para a direita, colocando na parte inferior da pilha de cada linha as cargas cujo destino é mais distante. A Aplicação desta regra considerando a matriz T da Figura 22 e que o navio se encontra no porto 2, resultará na matriz B da Figura 23(b).

- Regra Rc3: Esta regra é o espelho da regra Rc1, isto é, no porto p a matriz de ocupação B será preenchida por baia, da direita para a esquerda, colocando na parte inferior da pilha de cada baia as cargas cujo destino é mais distante. A Figura 23(c) ilustra o resultado obtido com esta regra considerando a matriz T da Figura 22 e o navio no porto 2.

- Regra Rc4: Esta regra é o espelho da regra Rc2, isto é, no porto p a matriz de ocupação B será preenchida por linha, da direita para a esquerda, colocando na parte inferior da pilha de cada baia as cargas cujo destino é mais distante. A Figura 23(d) ilustra o resultado obtido com esta regra considerando a matriz T da Figura 22 e que o navio no porto 2.

- Regra Rc5: Nesta regra o preenchimento da matriz de ocupação B em um porto p é feito por baia até uma linha p, começando pela coluna da esquerda e colocando-se em primeiro lugar os contêineres cujos destinos são os portos mais distantes. A linha p é calculada pegando-se a função teto, resultante da soma do total de contêineres que estavam no navio e foram embarcados nos portos anteriores; menos a quantidade de contêineres que serão desembarcados em p, mais a quantidade de contêineres que a serem embarcados em p, dividido pelo número de baias da matriz B. O número dessa linha pode ser calculado sobre a matriz de transporte T, através da Equação (10).

$$\theta_p = \left\lceil \frac{\sum_{i=1}^{p} \sum_{j=p+1}^{N} T_{ij}}{D \times C} \right\rceil \tag{10}$$

onde: p é o porto atual do navio, T_{ij} é o número total de contêineres a serem embarcados no porto i com destino ao porto j e D é o número total de baias da matriz B de ocupação do navio.

Supondo que a matriz de ocupação B quando o navio chega ao porto 2 seja a da Figura 23(b). Depois, os contêineres cujo destino é o porto 2 são descarregados e o navio move-se para o porto 3, e a regra de carregamento Rc5 é aplicada resultando na matriz de ocupação na Figura 23(e).

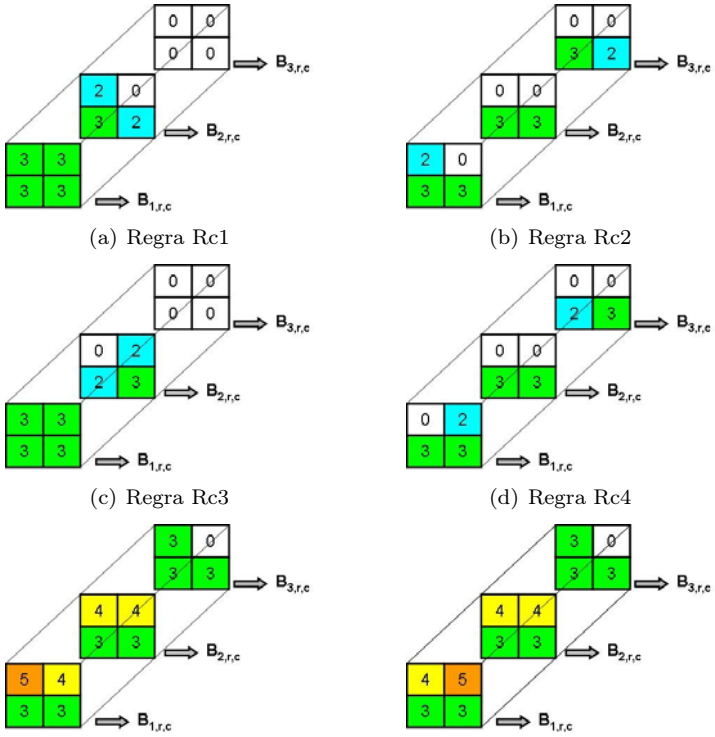
- Regra Rc6: Esta regra também é o espelho da regra Rc5. Ela faz o preenchimento da matriz de ocupação B em um porto p preenchendo cada coluna até a linha p, começando pela coluna da direita e colocando-se em primeiro lugar os contêineres cujos destinos são os portos mais distantes. A linha p é calculada pela Equação (10), de modo idêntico ao calculado na regra Rc5. Depois, os contêineres cujo destino é o porto 2 são descarregados e o navio move-se para o porto 3, e a regra de carregamento Rc5 é aplicada resultando na matriz de ocupação da Figura 23(f).

4.2.2 Regras de Descarregamento

- Regra Rd1: Nesta regra quando o navio chega a um porto p, são removidos todos os contêineres cujo destino é p e todos os contêineres que estão acima dos contêineres do porto p e cujos destinos são os portos $p + j$, para $j = 1, \cdots, N$ tal que $(p + j) \leq N$. Suponha por exemplo que ao se chegar no porto 2 a matriz de Ocupação B seja a dada na Figura 24(a). De acordo com esta regra a matriz B ficaria como mostrada na Figura 24(b). Além disso, após a aplicação da regra a matriz de transporte T é atualizada como na Figura 24(c).

- Regra Rd2: Nesta regra quando o navio chega ao porto p, todos os contêineres são removidos para permitir que todas as pilhas sejam reordenadas por alguma regra de carregamento a ser aplicada posteriormente. Assim, os contêineres removidos com destinos a portos além do porto 2 são contabilizados na matriz de transporte T, gerando a nova matriz de transporte tal como dado na Figura 24(d).

O leitor interessado na elaboração e o funcionamento de uma regra pode olhar os códigos elaborados em Matlab para as regras de entrada Re1 e de saída Rs1 que estão disponíveis na apêndice A.2.

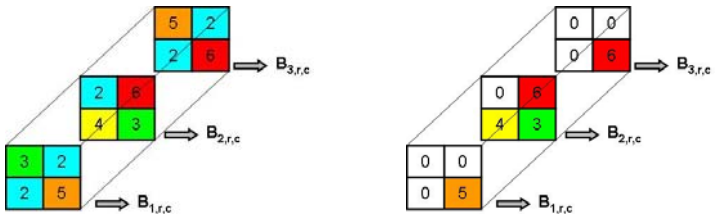


(a) Regra Rc1 (b) Regra Rc2

(c) Regra Rc3 (d) Regra Rc4

(e) Regra Rc5 (f) Regra Rc6

Figura 23: A matriz de estado B associada a arrumação dos contêineres no navio no porto 2.



(a) Antes da Regra Rd1 (b) Depois da Regra Rd1

	D2	D3	D4	D5	D6
O2	+0	+3	+0	+2	+0

(c) Matriz transporte após Rd1

	D2	D3	D4	D5	D6
O2	+0	+2	+1	+3	+1

(d) Matriz transporte após Rd2

Figura 24: A matriz de estado B associada a arrumação dos contêineres no navio no porto 2.

4.3 Avaliação do número de movimentos e instabilidade

As regras, que são combinações de regras de carregamento e descarregamento ilustradas nas seções anteriores, podem fornecer o número de movimentos realizados e a instabilidade do navio em um dado porto. Assim, após a aplicação

de uma sequência de regras em um navio até um certo porto é possível determinar o número de movimentos e a instabilidade do navio até aquele porto. Para tanto, é necessário empregar um procedimento de simulação que transforma as regras em movimentos dos contêineres dentro do navio. Tal procedimento de simulação é dado na Tabela 3.

Avaliação de uma Solução
Início
$p \leftarrow 1$, $nmov \leftarrow 0$, $instab \leftarrow 0.0$
inicializar(B, T)
enquanto $p < N$ **faça**
Início
$[rc, rd] =$ **extrairRegras**$(s(p))$
Se $(p > 1)$
$instab \leftarrow instab+$ **calcDXDZ**(B)
$[aux, B, T] =$ **descarregar**(rd, B, p)
$nmov \leftarrow nmov + aux$
fim
Se $(p < N - 1)$
$[aux, B, T] =$ **carregar**(rc, B, T, p);
$instab \leftarrow instab+$ **calcDXDZ(B)**
$nmov \leftarrow nmov + aux$
fim
$p \leftarrow p + 1$
retornar $\alpha \times nmov + \beta \times instab$
fim
fim

Tabela 3: Detalhamento da avaliação de uma solução do PCCTP 3D através de simulação.

Os símbolos e funções empregados no procedimento da Tabela 3 são descritos a seguir:

p - Variável contadora, que indica o porto atual da simulação.

N - Número total de portos.

S - Vetor tal que o elemento $s(i)$ contém a regra k, a ser aplicada no porto i e modificar a matriz B adequadamente.

$Nmov$ - Número de movimentos realizados para carregar ou descarregar o navio ao longo dos N portos.

B - Matriz de ocupação que indica o estado do navio em cada porto i.

rc - Variável que contém o nome da regra de carregamento a ser aplicada.

rd - Variável que contém o nome da regra de descarregamento a ser aplicada.

$inicializar$ - Função que preenche a matriz B, com valores iguais a zero.

extrairRegras - Função que define a correspondência entre a regra k, contida em $s(i)$, com as regras de descarregamento e carregamento a serem armazenadas nas variáveis rd e rc, respectivamente.

descarregar - Função que aplica a regra de descarregamento contida em rd na matriz B no porto i, e retorna o número de movimentos realizados, e B e T atualizadas.

carregar - Função que aplica a regra de carregamento contida em rc, na matriz B, no porto i e retorna o número de movimentos realizados e B e T atualizadas.

calcDXDZ - Função que realiza o cálculo da distância entre o centro de massa e o centro geométrico por baia d, para o navio, antes de ser descarregado e depois de ser carregado em cada porto p.

As funções *carregar* e *descarregar*, empregadas no procedimento da Tabela 3, utilizam as regras descritas nas seções 4.2.1 e 4.2.2, respectivamente, e fornecem o número de movimentos realizados (dado em *aux*). A instabilidade do navio é calculada através da função *calcDXDZ*. Para melhor ilustrar o funcionamento da função calcDXDZ, suponha que o navio tenha o arranjo de contêineres ,tal como ilustrado na Figura 25.

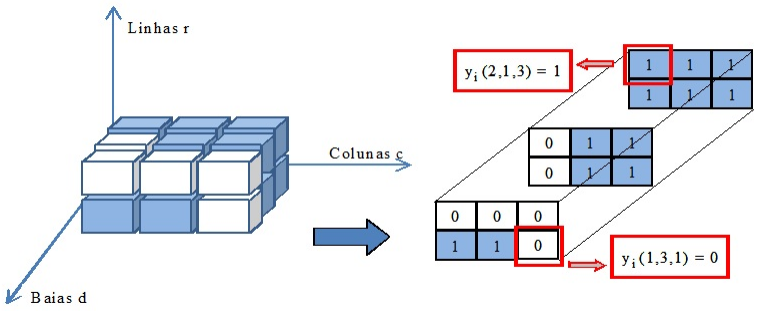


Figura 25: Ocupação do navio porta-contêiner e sua matriz de estado B correspondente.

Define-se xcm_3 como a coordenada do centro de massa relativa às linhas da baia e zcm_3 a coordenada do centro de massa relativa às colunas da baia 3. Os valores xcm_3 e zcm_3 podem ser calculados, usando a Eq. (9) e a informação contida na Figura 25, como dado na Figura 26.

$$xcm_3 = \frac{(2 \times 0,5 + 2 \times 1,5 + 2 \times 2,5)}{(2+2+2)} = \frac{9}{6} = 1,5$$

$$zcm_3 = \frac{(3 \times 0,5 + 3 \times 1,5)}{(6)} = \frac{6}{6} = 1,0$$

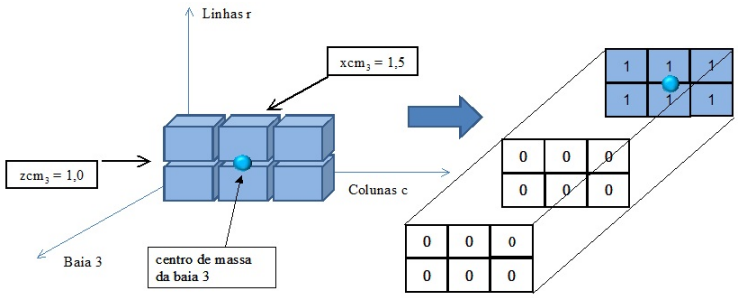


Figura 26: Cálculo das coordenadas do centro de massa da baia 3.

Cálculos semelhantes podem ser realizados para a baia 2, para a obtenção de xcm_2 e zcm_2 empregando-se a Eq. (9) e a informação contida na Figura 27.

$$xcm_2 = \frac{(0 \times 0,5 + 2 \times 1,5 + 2 \times 2,5)}{(0 + 2 + 2)} = \frac{8}{4} = 2,0$$

$$zcm_2 = \frac{(2 \times 0,5 + 2 \times 1,5)}{(4)} = \frac{4}{4} = 1,0$$

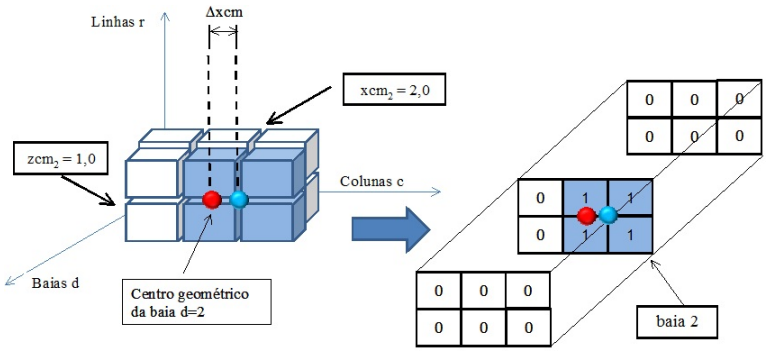


Figura 27: Cálculo das coordenadas do centro de massa da baia 2.

Finalmente, para a baia 1, as coordenadas xcm_1 e zcm_1 podem ser calculadas, usando-se a Eq. (9) e a Figura 28, como dado a seguir.

$$xcm_1 = \frac{(1 \times 0,5 + 1 \times 1,5 + 0 \times 2,5)}{(1 + 1 + 0)} = \frac{2}{2} = 1,0$$

$$zcm_1 = \frac{(2 \times 0,5 + 0 \times 1,5)}{(2 + 0)} = \frac{1}{2} = 0,5$$

De posse dos cálculos anteriores, é possível obter a medida de instabilidade para o navio, empregando a Eq. (9) e a configuração de contêineres apresentada na Figura 25 e como dado a seguir.

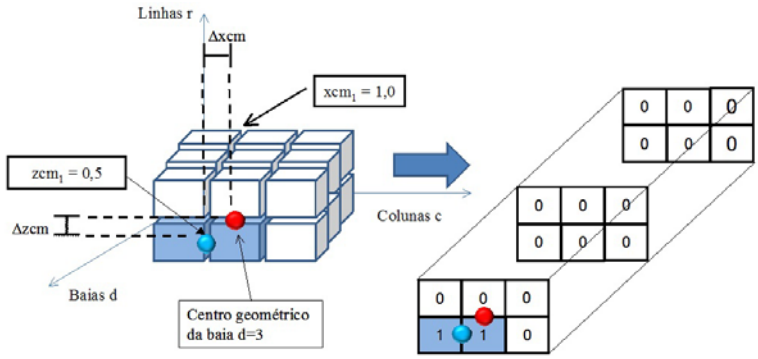


Figura 28: Cálculo das coordenadas do centro de massa da baia 1.

$$\phi_2(y) = \left[(1,5-1,5)^2 + (2,0-1,5)^2 + (1,0-1,5)^2\right] + \left[(1,0-1,0)^2 + (1,0-1,0)^2 + (0,5-1,0)^2\right] = 0,75$$

O leitor interessado nos detalhes de implementação das funções anteriores pode olhar os códigos elaborados em Matlab para as regras de entrada Re1 e de saída Rs1 que estão disponíveis na apêndice A.2.

4.4 Avaliação de uma Solução por Regras (Vetor de Regras)

Suponha que as dimensões do navio são tais como a que são apresentadas na Figura 21(a), isto é, 3 baias, 2 linhas 2 e 2 colunas. Os dados da matriz de transporte T são dados na Tabela 4.

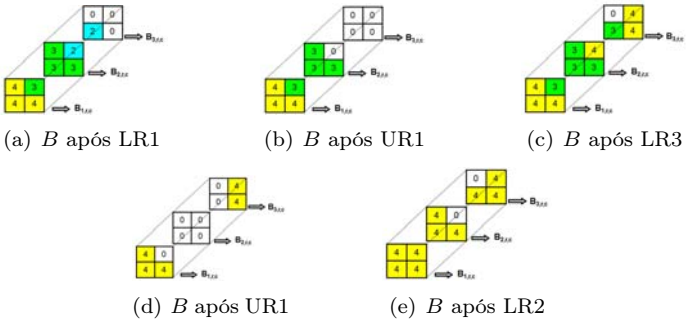
Tabela 4: Dados para a matriz de transporte T.

	$D2$	$D3$	$D4$
$O1$	2	4	3
$O2$	0	1	3
$O3$	0	0	4

Cada elemento $T(i,j) = nc$ da Tabela 4 indica que nc contêineres deverão embarcar no porto (linha) i para o porto (coluna) $j+1$. Por exemplo, o elemento $T(2,3) = 3$ significa que, no porto 2, 3 contêineres, cujo destino é o porto 4, deverão embarcar.

Agora, suponha que um vetor $v = [153]$ é proposto para este problema e que deseja-se avaliar o mesmo em termos do número de movimentos e da medida de instabilidade a partir do procedimento de simulação descrito na Tabela 3. Neste caso, tal procedimento funcionaria tal como descrito a seguir.

- *Porto 1 - Carregamento:*



(a) B após LR1 (b) B após UR1 (c) B após LR3

(d) B após UR1 (e) B após LR2

Figura 29: A matriz de estado B associada a arrumação dos contêineres no navio após a aplicação das regras.

De acordo com o vetor de solução v, $v[1] = 1$, ou seja, para o porto 1 a regra 1 deve ser aplicada tal que, de acordo com a Tabela 2, a regra de carregamento $LR1$ deve ser usada como ilustrado na Figura 29(a). Após esta operação, as variáveis que medem o número de movimentos e a instabilidade terão os seguintes valores:

- $instab = 2 \times (0.5)^2 = 0.5$,
- $nmov = 9$.

- *Porto 2 - Descarregamento:*

  De acordo com o vetor de solução v, $v[2] = 5$, ou seja, para o porto 2 a regra 5 deve ser aplicada tal que, de acordo com a Tabela 2, a regra de carregamento $UR1$ deve ser usada como ilustrado na Figura 29(b).

  Após esta operação, as variáveis que medem o número de movimentos e a instabilidade terão os seguintes valores:

  - $instab = 0,5 + (2 \times 0,166^2 + 2 \times 1,0) = 0,5 + (0,055 + 2,0) = 2,555$,
  - $nmov = 9 + 2 = 11$.

- *Porto 2 - Carregamento:*

  De acordo com o vetor de solução v, $v[2] = 5$, ou seja, para o porto 2 a regra 5 deve ser aplicada tal que, de acordo com a Tabela 2, a regra de carregamento $LR3$ deve ser usada como ilustrado na Figura 29(c).

  Após esta operação, as variáveis que medem o número de movimentos e a instabilidade terão os seguintes valores:

  - $instab = 2,555 + (2 \times 0,166^2) = 2,555 + 0,055 = 2,610$,
  - $nmov = 11 + 4 = 15$.

- *Porto 3 - Descarregamento:*

  De acordo com o vetor de solução v, $v[3] = 3$, ou seja, para o porto 3 a regra 3 deve ser aplicada tal que, de acordo com a Tabela 2, a regra de carregamento $UR1$ deve ser usada como ilustrado na Figura 29(d).

  Após esta operação, as variáveis que medem o número de movimentos e a instabilidade terão os seguintes valores:

- $instab = 2,610 + (2 \times 0,166^2 + 2 \times 1,0 + 0,5^2) = 4,91$,
- $nmov = 15 + 6 = 21$.

- *Porto 3 - Carregamento:*

  De acordo com o vetor de solução v, $v[3] = 3$, ou seja, para o porto 3 a regra 3 deve ser aplicada tal que, de acordo com a Tabela 2, a regra de carregamento $LR2$ deve ser usada como ilustrado na Figura 29(e).

  Após esta operação, as variáveis que medem o número de movimentos e a instabilidade terão os seguintes valores:

  - $instab = 4,91 + (2 \times 0,166^2 + 2 \times 0,166^2) = 5,57$,
  - $nmov = 21 + 5 = 26$.

- *Porto 4 - Descarregamento:*

  Para o último porto, existe apenas uma opção que é o descarregamento de contêineres do navio, e esta ação produz um número de movimentos igual a quantidade de contêineres cujo destino é o porto 4. Neste caso, os novos valores de instabilidade e número de movimentos serão:

  - $instab = 4,91 + (2 \times 0,166^2 + 2 \times 0,166^2) = 5,57$,
  - $nmov = 21 + 5 = 26$.

A avaliação de um solução representada por regras, nos moldes do que foi descrito anteriormente, será empregada pelas meta-heurísticas e heurísticas descritas nas próximas seções. Tendo em vista que a principal contribuição do trabalho é o emprego da representação por regras e seu acoplamento com meta-heurísticas, não foram realizados estudos detalhados sobre os melhores parâmetros a serem empregados por cada método. Assim, foi escolhido para cada método as opções mais rápidas e fáceis de serem codificadas, pois o propósito é mostrar que mesmo nestes casos, um resultado satisfatório pode ser obtido em pequeno tempo computacional. Deste modo, por exemplo, foi escolhido para o AG o método da roleta devido a sua facilidade de implementação.

4.5 Abordagem via Algoritmo Genético, *Simulated Annealing* e *Beam Search*

Um Algoritmo Genético mantém uma população de indivíduos $A(t) = \{A_{1t}, \cdots, A_{nt}\}$ na iteração (geração) t e cada indivíduo representa um candidato à solução do problema em questão. Na implementação computacional aqui utilizada, o indivíduo é representado através da coluna A_i, da matriz A Holland (1975); Michalewicz (1996); Man et al. (1999); Sivanandam & Deepa (2007). Cada solução A_{it} é avaliada e produz alguma medida de adaptação, ou *fitness*, então, uma nova população é formada na iteração $t+1$ pela seleção dos indivíduos mais adaptados. Alguns indivíduos da população são submetidos a um processo de alteração por meio de operadores genéticos para formar novas soluções. Existem transformações unárias m_i (mutação) que criam novos indivíduos através de pequenas modificações de atributos em um indivíduo ($m_i : A_i \rightarrow A_i$), e transformações de ordem superior c_j (*crossover*), que criam novos indivíduos através da combinação de dois ou mais indivíduos ($c_j : A_j ... A_k \rightarrow A_j$). Após

um número de gerações, a condição de parada deve ser atendida, seja porque existe, na população, um indivíduo que represente uma solução aceitável para o problema, seja porque o número máximo de gerações foi atingido.

O Algoritmo Genético deste capítulo combina a representação de soluções com o esquema de regras descrito na seção 4.2. Com isto os indivíduos são codificados em uma notação compacta que sempre fornece soluções factíveis. Esta estratégia permite o tratamento de problemas de maior porte em tempo computacional razoável. Maiores detalhes sobre o Algoritmo Genético para o PCCTP 3D é dado em Azevedo et al. (2011b, 2014).

A metaheurística *Simulated Annealing* (SA) é um algoritmo de busca local capaz de escapar de ótimos locais, permitindo movimentos para soluções com valor da função objetivo piores que o valor da função objetivo da solução corrente. A facilidade de implementação, as propriedades de convergência e o uso de movimentos de " subida " (*hill-climbing*) com o objetivo de escapar de ótimos locais contribuíram para que se tornasse uma técnica muito popular em Otimização Combinatória Henderson et al. (2003). As origens do algoritmo se fundamentam na Mecânica Estatística, a idéia da metaheurística tendo sido inspirada no processo de recozimento de metais e vidro, o qual assume uma configuração de baixa energia quando inicialmente aquecido e então resfriado lentamente Aguilera et al. (2008); Talbi (2009). O algoritmo SA foi inicialmente apresentado como um algoritmo de busca para problemas de Otimização Combinatória em Kirkpatrick et al. (1983). Detalhes sobre o SA para o PCCTP 3D são fornecidos em Azevedo et al. (2011a, 2014).

O algoritmo do *Beam Search* (BS) é um método do tipo Enumeração Implícita para resolver problemas de Otimização Combinatória Sabuncuoglu & Baviz (1999); Croce & T'Kindt (2002); Fox (1983); Valente & Alves (2005); Ribeiro & Azevedo (2010). Pode-se dizer que ele é uma adaptação do método de *Branch and Bound* onde somente os nós mais promissores de cada nível da árvore de decisões (atribuições) são guardados na memória para serem visitados, enquanto que os demais nós são descartados permanentemente.

Como uma grande parte dos nós da árvore de atribuições é descartada, sem ser analisado, o tempo de execução do BS é polinomial com relação ao tamanho do problema. Em resumo pode-se dizer que o BS é uma técnica de busca em árvore de decisão que em cada nível da árvore é analisado um número fixo de nós e, por conseguinte, um número fixo de soluções. Maiores detalhes acerca da implementação do BS para o problema PCCTP 3D podem ser encontrados em Azevedo et al. (2012).

4.6 Resultados Obtidos

Para testar o algoritmo desenvolvido, foram geradas automática e aleatoriamente 15 instâncias. Essas instâncias são classificadas de acordo com o número de portos e o tipo da matriz de transporte. Para cada instância é gerada uma matriz de transporte T, tal que a capacidade do navio não será excedida em nenhum porto, isto é, o valor de p dado pela equação (11) deve ser menor ou igual a $D \times R \times C$ para todo porto p, isto porque a matriz de transporte é factível desde que a Eq. (11) seja satisfeita.

$$\sum_{i=1}^{p}\sum_{j=p+1}^{N} \leq D \times R \times C, \forall p = 1, \cdots, N. \quad (11)$$

De acordo com Avriel et al. (1998) podem ser gerados três tipos de matriz de transporte: 1-Mista, 2-Longa distância e 3-Curta distância. Uma matriz do tipo 3 se refere ao transporte de contêineres que vão percorrer poucos portos antes de serem desembarcados. Já uma matriz do tipo 2 se refere a contêineres que vão percorrer muitos portos antes de serem desembarcados. Uma matriz do tipo mista combina os dois tipos anteriores. As instâncias foram classificadas de acordo com a quantidade de portos a serem percorridos, o tipo da matriz de transporte e a capacidade do navio. Neste trabalho foi suposto um navio com as seguintes dimensões ($D \times R \times C$): $5 \times 6 \times 50$, resultando na seguinte capacidade máxima: 1500 contêineres. As instâncias utilizadas nos testes encontram-se disponíveis em:

`https://sites.google.com/site/3dcontainershipproject/home.`

As Tabelas 5, 6 e 7 mostram os resultados obtidos com a melhor solução após 5 rodadas de AG, 5 rodadas de SA e 1 rodada do BS para as 15 instâncias, respectivamente. As tabelas possuem as colunas relacionadas a seguir: I: o número da instância; M: o tipo da matriz de transporte; N: o número de portos; $NMin$: o número mínimo de movimentos a serem realizados com os contêineres; $F.O.1$: os valores da função objetivo em termos do número total de movimentos realizados com os contêineres até a chegada no último porto (como dado pela Eq. (8)); $F.O.2$: os valores de instabilidade de acordo com o arranjo de contêineres e com a Eq. (9); e T: tempo computacional em segundos.

Note-se que os valores de $F.O.1$, $F.O.2$ e T são apresentados para dois pares de α e β: $(\alpha = 1, \beta = 0)$ e $(\alpha = 0, \beta = 1)$. Note-se ainda que os resultados apresentados para $(\alpha = 1, \beta = 0)$ correspondem à melhor solução obtida com o objetivo de minimizar o número de movimentos para cada instância e que para 7 instâncias o número de movimentos ($FO1$) encontrado é até 2% maior que o número mínimo ($Nmin$), para 5 instâncias o número mínimo é até 10% maior e apenas 3 instâncias até 14%. Já para $(\alpha = 0, \beta = 1)$ como o objetivo é minimizar a medida de instabilidade, apresenta maior número de movimentos ($FO1$), mas soluções com medida de instabilidade significativa menor ($FO2$). Estes dados confirmam que a minimização do número de movimentos e da instabilidade são objetivos conflitantes. O número mínimo de movimentos é obtido multiplicando-se por dois o valor do somatório de T_{ij} calculado de acordo com a equação (8). Observe-se que esse valor é o limitante inferior para o número total de movimentos a serem realizados ao longo do percurso do navio. Os resultados de todos os métodos foram obtidos com um programa desenvolvido em Matlab 7.0 e executado num computador Intel Core 2Duo 2.20 GHz (E4500), 2GB RAM, Windows XP Versão 2002 (SP 3).

A primeira observação importante acerca dos resultados das Tabelas 5, 6, 7 é sobre o tempo computacional gasto pelos métodos. É importante observar que para a formulação dada pelas equações (2)-(9) as instâncias com 30 portos são problemas tais que uma única solução deve ser representada por 40.545.000 variáveis inteiras (30 portos, 5 baias, 6 linhas e 50 colunas). Para estas instâncias o SA, por exemplo, consegue produzir boas soluções em menos de 1 hora. Pode-se observar também que, de forma geral, um aumento de 5 no número de

Tabela 5: Resultados do AG para dois pares de valores de α e β.

I	N	M	Nmin	$\alpha = 1$ e $\beta = 0$			$\alpha = 0$ e $\beta = 1$		
				FO1	FO2	T	FO1	FO2	T
1	10	1	6994	7072	564,45	146	10106	39,12	152
2	10	2	4172	4214	542,88	139	6374	13,88	145
3	10	3	17060	17116	566,89	149	18018	63,31	153
4	15	1	9974	10584	237,62	234	12766	30,01	241
5	15	2	4824	5030	514,54	210	9078	36,12	232
6	15	3	24902	25046	578,56	221	25844	113,35	258
7	20	1	10262	10802	229,31	300	14658	77,21	360
8	20	2	4982	5500	439,95	277	9784	17,13	317
9	20	3	32602	32638	792,20	279	33700	488,88	330
10	25	1	11014	11848	252,34	353	17302	19,02	558
11	25	2	5002	5466	573,02	322	10564	98,93	440
12	25	3	43722	44082	659,70	387	44964	161,29	455
13	30	1	11082	12580	188,38	476	16556	32,45	513
14	30	2	4720	5312	156,04	381	7098	10,43	402
15	30	3	53592	54398	427,08	461	56290	166,05	472

Tabela 6: Resultados do SA para dois pares de valores de α e β.

I	N	M	Nmin	$\alpha = 1$ e $\beta = 0$			$\alpha = 0$ e $\beta = 1$		
				FO1	FO2	T	FO1	FO2	T
1	10	1	6994	7068	507,14	1654	11122	14,20	1385
2	10	2	4172	4208	492,45	1531	6672	11,55	1318
3	10	3	17060	17088	583,18	1431	17554	92,85	1094
4	15	1	9974	10420	221,40	2054	13910	30,10	1850
5	15	2	4824	5082	532,23	2044	8098	34,49	1761
6	15	3	24902	24998	595,22	1887	25896	154,01	1607
7	20	1	10262	10749	202,09	2485	16396	13,60	2286
8	20	2	4982	5458	451,71	2482	10022	14,57	2258
9	20	3	32602	32632	1000,11	2554	33378	575,36	2299
10	25	1	11014	11590	200,89	2866	14674	12,69	2767
11	25	2	5002	5430	561,69	2495	9456	106,50	2499
12	25	3	43722	44078	543,47	2624	45120	218,75	2727
13	30	1	11082	12146	183,97	3158	18180	17,99	3158
14	30	2	4720	5246	168,66	2687	8604	13,99	2708
15	30	3	53592	54454	519,62	3076	55754	299,82	3082

Tabela 7: Resultados do BS para dois pares de valores de α e β.

I	N	M	Nmin	$\alpha = 1$ e $\beta = 0$			$\alpha = 0$ e $\beta = 1$		
				FO1	FO2	T	FO1	FO2	T
1	10	1	6994	7072	588,45	429	10432	17,59	507
2	10	2	4172	4202	478,22	403	7172	12,52	522
3	10	3	17060	17074	577,54	453	17642	15,83	618
4	15	1	9974	10234	194,24	1752	15058	12,59	2872
5	15	2	4824	4936	603,19	1550	9560	9,93	2274
6	15	3	24902	24992	420,96	1808	25952	17,38	2022
7	20	1	10262	10432	163,64	4275	16374	6,68	5723
8	20	2	4982	5152	447,15	3946	8652	10,21	4863
9	20	3	32602	32610	1146,09	4102	33544	431,16	4651
10	25	1	11014	11154	285,08	8217	14058	4,59	9684
11	25	2	5002	5156	553,64	7576	10240	92,11	9455
12	25	3	43722	43942	613,83	9506	45304	32,45	10397
13	30	1	11082	11430	198,93	15756	18972	6,04	18073
14	30	2	4720	5598	126,63	14468	7544	6,59	15066
15	30	3	53592	53896	701,56	17335	55062	29,81	18301

portos a serem percorridos, de uma instância para outra, produz em média um aumento de mais ou menos 7 minutos no tempo computacional gasto pelo SA. Por exemplo, em instâncias com 10 portos leva-se 27 minutos e 30 segundos para se obter uma solução, ao passo que em instâncias com 15 portos leva-se, em média, 34 minutos. O BS apesar de ter apresentado os maiores tempos computacionais também foi o que apresentou os melhores resultados para a maioria das instâncias e para as duas funções objetivo. Espera-se, ainda, que futuras implementações em linguagem C venham a reduzir o tempo computacional de solução para todos os métodos.

Os resultados indicam que para as instâncias em que a função objetivo visa minimizar o número de movimentos ($\alpha = 1, \beta = 0$) e a matriz de transporte é do tipo curta distância (tipo 3) as regras são bastante adequadas e produzem resultados muito próximos do limite inferior do número de movimentos, e chegam mesmo a quase atingir este limite, como no caso para 10, 15 e 20 portos produzem soluções com distância de $0,16\%$, $0,32\%$ e $0,09\%$, respectivamente. Já para as instâncias em que a matriz de transporte é do tipo média distância (tipo 1) e longa distância (tipo 2), todos os métodos apresentaram soluções cujo número de movimentos é ligeiramente maior que o limitante inferior. Estes resultados indicam a necessidade de se incorporar ao sistema um número maior de regras que levem em consideração a arrumação de contêineres que permanecerão um longo período de tempo dentro do navio. Porém, quando a função objetivo visa minimizar a medida de instabilidade ($\alpha = 0, \beta = 1$) as soluções apresentadas por todos os métodos têm um razoável aumento no número de movimentos, mas com uma significativa melhora na medida de instabilidade. Em algumas instâncias essa medida pode ser até 50 vezes menor em comparação com a solução encontrada na minimização do número de movimentos (instância 2).

As Figuras 30, 31 e 32 ajudam a melhor ilustrar esse *tradeoff* entre as duas funções objetivo, bem como a natureza bi-objetivo do problema.

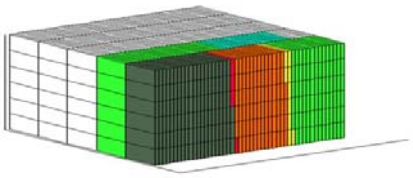


Figura 30: Disposição dos contêineres no navio para o porto 2 para a melhor solução encontrada com o AG, o SA e o BS para $(\alpha = 1, \beta = 0)$.

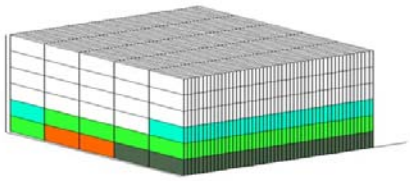


Figura 31: Disposição dos contêineres no navio para o porto 2 para a melhor solução encontrada com o AG e o SA para $(\alpha = 0, \beta = 1)$.

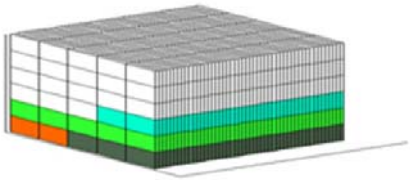


Figura 32: Disposição dos contêineres no navio para o porto 2 para a melhor solução encontrada com o BS para $(\alpha = 0, \beta = 1)$.

5 Integrando o Plano de Estiva com o *Scheduling* de Guindastes Portuários

Conforme visto na seção 2, de acordo com Guan et al. (2013); D. Steenken (2004), as operações em um terminal portuário de contêineres podem ser divididas em cinco problemas principais:

1. Alocação de berços: a saída deste problema é a programação de qual navio será atendido em qual berço de modo que uma distância mínima de segurança seja respeitada entre dois navios, e dois navios não podem compartilhar o mesmo berço no mesmo período de tempo;

2. Plano de estiva: este problema consiste em determinar como organizar os contêineres em um navio de modo a minimizar o número de movimentos de carregamento e descarregamento;

3. Designação e *Scheduling* de Guindastes no Cais: o tempo que o navio porta-contêiner leva para carregar ou descarregar depende da programação de guindastes portuários alocados para cada seção do navio. Os guindastes não podem ultrassapar uns aos outros por terem seus movimentos limitados a um trilho comum. Além disso, uma distância de segurança entre eles também deve ser observada;

4. Transporte do Cais: é necessário determinar que máquinas serão empregadas e quais serão suas trajetórias ligando o navio ao pátio do porto e vice-versa;

5. Transporte do Pátio: A organização de pilhas no pátio do porto depende de como realizar operações eficientes para empilhar e desempilhar os contêineres que são trazidos ou levados por caminhões e trens.

Esta seção irá discutir e propor um método para a obtenção de um solução conjunta dos problemas (2) e (3) dentro da perspectiva lançada por Zeng & Yang (2009):

> "Many complex systems such as manufacturing, supply chain, and container terminals are too complex to be modeled analytically. Discrete event simulation has been a useful tool for evaluating the performance of such systems. However, simulation can only evaluate a given design, not provide more optimization function. Therefore, the integration of simulation and optimization is need."

Vale observar, que existe uma tendência recente de se procurar fornecer soluções que integrem os cinco principais problemas encontrados em um porto. Um exemplo disto é fornecido nos seguintes artigos: consideração integrada dos problemas de alocação de berços e *scheduling* de guindastes portuários (Bierwirth & Meisel, 2010; Chang et al., 2010; Imai et al., 2008; Meisel & Bierwirth, 2009; Yang et al., 2012), integração entre alocação de berços e planejamento da operação do pátio (Hendriks et al., 2013), e alocação de contêineres vazios no pátio com o problema de roteamento de veículos (Braekers et al., 2013).

A perspectiva empregada neste texto para lidar com a integração de problemas encontrados no porto é reforçada pelo fato de que, recentemente, a prestigiosa revista *Transportation Research Part E: Logistics and Transportation Review* dedicou um volume especial a integração entre simulação e otimização em sistemas logísticos (*Special Issue on: Modeling, Optimization and Simulation of the Logistics Systems, volume 65, Pages 1-98, May 2014*).

5.1 O *Scheduling* de Guindastes Portuários

Para realizar a integração do Plano de Estiva com o *Scheduling* de Guindastes Portuários, é necessário observar que o plano de estiva guarda relação estreita com a programação de guindastes, pois a última consiste em determinar a posição dos guindastes portuários nas baias do navio ao longo do horizonte de planejamento. A Figura 33 ilustra melhor esse relacionamento.

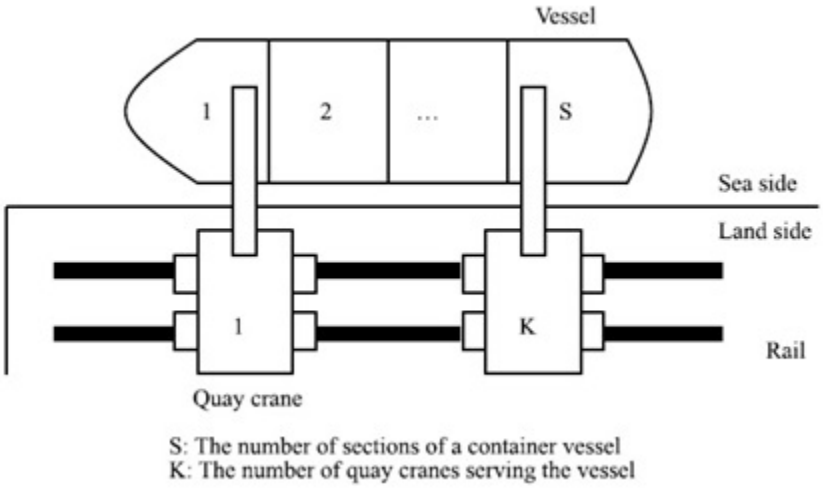


Figura 33: Problema de programação de guindastes portuários (Fonte: Lee et al. 2008).

Para evidenciar a complexidade de realizar a otimização conjunta de tais problemas, na literatura são encontradas ou abordagens que tratam do problema de estiva (Ambrosino et al., 2006, 2010; Avriel et al., 1993, 1998; Botter & Brinati, 1991; Dubrovsky et al., 2002; Fan et al., 2010; A. Imai et al., 2006; Wilson & Roach, 2000) ou somente do problema de programação de guindastes portuários tal que a informação do número de contêineres a serem descarregados ou carregados, por baia, é dada (Chung & Choy, 2012; Guan et al., 2013; Homayouni et al., 2011; Javanshir & Ganji, 2010; Kim & Park, 2004; Lee et al., 2008; Legato et al., 2008; K. Mak, 2009; Tavakkoli-Moghaddam et al., 2009). Isto, porque o problema de *scheduling* de guindastes portuários, embora possa parecer com o problema de m máquinas paralelas possui características adicionais que o tornam mais complicado tal como observado por Kim & Park (2004):

> "Similar to *m-parallel machine* scheduling problem (e.g., Guinet, 1993; Lee and Pinedo, 1997), the QC scheduling problem in this study address scheduling multiple QCs to perform discharging and loading operations in order to minimize the makespan of the operation. However, the QC scheduling problem has several unique characteristics that are different from those of a typical *m-parallel machine* problem. For example, when discharging and loading operations must be performed at the same ship-bay, the discharging must precede the loading operation. $(\cdots)$. Also, it should be noted that QCS travel on the same track. Thus, certain pairs of tasks cannot be performed simultaneously when the locations of the two clusters corresponding to the tasks are too close to each other, because two adjacent QCs must be apart from each other by at least one ship-bay so that they can simultaneously perform their tasks without interference.".

Assim, a otimização conjunta do plano de estiva e *scheduling* de guindastes portuários é um problema desafiador. Neste sentido, Kim & Park (2004) formularam o problema como um modelo de programação inteira mista e observaram que os experimentos computacionais demandaram muito tempo para que o modelo possa ter qualquer uso prático. Os autores ilustram tal afirmativa na seguinte frase:

> "For example, a problem with two QCs and six tasks require 480 minutes to solve completely, using extended LINDO/PC (Release 6.1), which is a specialized software for integer and linear programs.".

Com o intuito de melhor descrever as características do problema de *scheduling* de guindastes portuários, foi adotado o modelo descrito em Guan et al. (2013). Cabe uma observação importante que a primeira consideração que foi alterada para poder se acoplar o modelo com o problema do plano de estiva, mas as demais foram mantidas sob a forma de restrições e função objetivo originalmente fornecidas:

- Ao invés de se considerar o navio dividido em seções como em Guan et al. (2013), foi considerada cada baia do navio. Dessa forma, é possível transformar a informação de movimentação de contêineres fornecida pelo plano de estiva sob a forma do total de esforço necessário por baia;
- Foi assumido que o esforço total equivale ao número de contêineres que devem ser movidos e que cada container demanda uma unidade de tempo para ser carregado ou descarregado;
- Cada baia no navio só pode ter um único guindastes em um dado período de tempo;
- Uma vez que um guindaste começou o serviço (carregamento ou descarregamento) em uma baia, o mesmo só irá parar quando não houver mais serviço para realizar;
- Quando um guindaste troca sua operação de uma baia para outra, ele leva um tempo constante e igual a três unidades de tempo para mover um contêiner;

- Uma distância minima entre os guindastes deve ser observada. Isto significa que, às vezes, um guindaste não pode ser movido e não pode começar o seu serviço até que um segundo guindaste termine o seu trabalho e mova-se para uma nova posição que garanta a distância mínima entre os guindastes;
- Um guindaste não deve ultrapassar outro, pois eles compartilham o mesmo trilho;
- Todos os guindastes possuem a mesma e constant taxa de serviço de movimentação de uma unidade de trabalho por unidade de tempo.

O modelo matemático em termos de programação linear com variáveis binárias para o problema de programação de Guindastes Portuários proposto por Guan et al. (2013) é dado pelas Eqs. (12)-(23).

$$\min \tau \tag{12}$$

sujeito a

$$tx_{j,t} \leq \tau, j = 1, \cdots, S; t = 1, \cdots, T \tag{13}$$
$$w_{j,t} - Mx_{j,t} \leq 0, j = 1, \cdots, S; t = 1, \cdots, T \tag{14}$$
$$w_{j,t-1} - \mu y_{i,j}^{t-1,t} \leq w_{j,t}, j = 1, \cdots, S; t = 1, \cdots, T \tag{15}$$
$$z_{j,j+1}^{t,t+1} + z_{j,j-1}^{t,t+1} + z_{j,j}^{t,t+1} = b_j, j = 1, \cdots, S; t = 1, \cdots, T \tag{16}$$
$$z_{j-1,j}^{t-1,t} + z_{j,j}^{t-1,t} + z_{j+1,j}^{t-1,t} - z_{j,j+1}^{t,t+1} - z_{j,j-1}^{t,t+1} - z_{j,j}^{t,t+1} = 0,$$
$$j = 1, \cdots, S; t = 2, \cdots, T-1 \tag{17}$$
$$z_{j-1,j}^{t-1,t} + z_{j,j}^{t-1,t} + z_{j+1,j}^{t-1,t} = z_{j,j+1}^{t,t+1}, j = 1, \cdots, S; t = 1, \cdots, T \tag{18}$$
$$\sum_{j=1}^{S} z_{j,1}^{t-1,t} = \sum_{j=1}^{S} b_j, t = T+1 \tag{19}$$
$$z_{j-1,j}^{t-1,t} + z_{j,j}^{t-1,t} + z_{j+1,j}^{t-1,t} \leq 1, j = 1, \cdots, S; t = 2, \cdots, T \tag{20}$$
$$z_{j-1,j}^{t-1,t} + z_{j,j}^{t-1,t} \leq 1, j = 1, \cdots, S; t = 2, \cdots, T \tag{21}$$
$$y_{j,j}^{t-1,t} - z_{j,j}^{t-1,t} \leq 0, j = 1, \cdots, S; t = 2, \cdots, T \tag{22}$$
$$y_{j,j}^{t-1,t} + x_{j,t} - y_{j,j}^{t,t+1} \leq 1, j = 1, \cdots, S; t = 2, \cdots, T \tag{23}$$
$$w_{j,t} \geq 0 \ \ with \ \ w_{j,1} = W_j, x_{j,t}, y_{j,j}^{t,t+1}, z_{j,j}^{t,t+1} \in [0,1].$$

Onde: M é um valor grande, t é o índice relativo aos períodos de tempo, T é o horizonte de tempo que pode ser qualquer valor maior que o Makespan τ, é uma ponderação acerca da carga de trabalho em uma dada seção j no período t, $b_j = 1$ se um guindaste está originalmente localizado na seção j e 0 caso contrário, W_j é a carga inicial de trabalho na seção j. Para o modelo dado pelas Eqs (12)-(23) as seguintes variáveis de decisão foram consideradas:

1. Uma variável binária $x_{j,t}$ tem valor 1, se ainda existir carga de trabalho na baia j no início do período t e 0, caso contrário;
2. A variável inteira $w_{j,t}$ representa a carga de trabalho remanescente na baia j no início do período de tempo t;

3. A variável inteira $z_{j,\tilde{j}}^{t,t+1}$ representa o número de guindastes que se movem da baia j para a baia $\tilde{j}$ durante o período de tempo t;

4. A variável inteira $y_{j,j}^{t,t+1}$ representa o número de guindastes na baia j durante o período de tempo t.

A função objetivo consiste na minimização do tempo necessário para terminar os serviços no navio, incluindo movimentos de carregamento e descarregamento, e mudança de posição do guindaste. Assim, a função objetivo (12) é a minimização do *makespan*. A restrição (13) determina que o tempo de término da última unidade de trabalho, i.e., o *makespan*. A restrição (14) estabelece a existência ou não da carga de trabalho restante na seção j no tempo t. A restrição (15) representa o balanceamento do fluxo de carga de trabalho da seguinte forma: para cada seção j existe uma carga inicial de trabalho W_j (dada pelo plano de estiva). Para cada período de tempo escolhido, a carga de trabalho remanescente será reduzida em unidades antes de atingir zero, se houver um guindaste portuário operando durante o período de tempo selecionado. As restrições (16) até (19) são relativas a conservação do fluxo de guindastes. A restrição (20) indica que para cada unidade de tempo, cada seção pode comportar no máximo um único guindaste. A restrição (21) indica que um guindaste não pode cruzar outro. A restrição (22) indica que o guindaste que um guindaste pode permanecer inativo em uma seção j durante um período t. Finalmente, a restrição (23) representa restrições de não-preempção, isto é, um guindaste não pode começar uma nova tarefa até que a tarefa atual para qual foi alocado esteja terminada.

Deve ser destacado que os modelos matemáticos são apresentados para ilustrar a complexidade e o modo com os dois problemas podem ser acoplados. Neste sentido, pela primeira vez na literatura, os dois modelos descritos anteriormente foram acoplados ao se utilizar baias ao invés de seções no problema dos guindastes. Dessa forma, foi possível empregar a observação de que o resultado do problema de estiva (P), o número de movimentos por baia, pode ser transformado na entrada do problema de operação de guindastes (O), o esforço total por baia, tal como dado na Figura 34.

É importante observar que ao realizar o acoplamento entre o plano de estiva (P) e a operação de guindastes (O), a resolução do problema (P) dependerá, em cada porto, da resolução do problema de guindastes (O). Assim, uma instância que considere um problema conjunto de (P) e (O) com um horizonte de 30 portos deverá resolver não só o problema (P) dado pelas Eqs. (1)-(8), mas, também, 60 sub-problemas (O) dado pelas Eqs. (9)-(21). Isto se deve ao fato de que, no problema conjunto, as operações de carregamento e descarregamento em cada porto devem ser realizadas através da programação e operação de guindastes portuários.

5.2 Método de solução via Representação por Regras

Como demonstrado em Avriel et al. (2000), o modelo matemático com variáveis binárias para o problema de estiva (P) não é adequado para problemas de grande porte. Caso a representação por variáveis binárias seja empregada, mesmo com

Figura 34: Acoplando o plano de estiva (P) com a operação de guindastes (O).

a adoção de meta-heurísticas, o número de variáveis a serem empregadas cresce de forma proibitiva. No artigo (Azevedo et al. 2012, 2014), ao invés de empregar uma meta-heurística que usa variáveis binárias, é empregada uma representação alternativa na resolução do problema de estiva (P) através integrada dos dois problemas através da representação por regras. Neste artigo, além de empregar a representação por regras, pela primeira vez, também para o problema de operação de guindastes (O), estas foram combinadas às regras do problema de estiva (P) para permitir a resolução integrada de ambos os problemas. A combinação entre as regras para carregamento e descarregamento de navios descritas em Azevedo et al. (2014) com quatro regras de operações de guindastes tal como dado na Tabela 8.

Na Tabela 8, RG representam as regras de alocação de guindastes, RC as regras de carregamento, RD as regras de descarregamento e R as regras finais que são combinações das demais. A Tabela 8 mostra apenas as primeiras 16 combinações das 64 possíveis de regras finais, pois foram utilizadas 4 regras de programação de guindastes. A Figura 35 ilustra o passo-a-passo do funcionamento de uma das regras de operação de dois guindastes portuários.

Apesar do uso da representação por regras reduzir consideravelmente a complexidade do problema a ser resolvido, a decisão de qual regra, das que estão descritas naTabela 1, deve ser aplicada em cada porto resulta em um problema combinatório. Para um problema que considera um horizonte de 30 portos, existem 64^{30}, ou seja, aproximadamente $1,54 \times 10^{54}$, possíveis soluções. Assim, emprega-se um algoritmo genético para resolver o problema combinatório resultante. A estrutura do algoritmo genético empregado neste trabalho é similar a descrita em Azevedo et al. (2014), com a única diferença em relação à codificação que mapeia as soluções e os indivíduos da população. Uma representação compacta da solução pode ser realizada através de um vetor S cujo elemento $s[j]$ de valor igual a k indica qual regra (daquelas fornecidas na Tabela 8) foi utilizada para o porto j em uma codificação de valores inteiros simular aquela que foi proposta para o problema do plano de estiva e tal como dado na Figura 36. Então, no exemplo da Figura 36, tem-se que no porto 1, foi utilizada a regra 4, no porto 2 a regra 3 e assim sucessivamente.

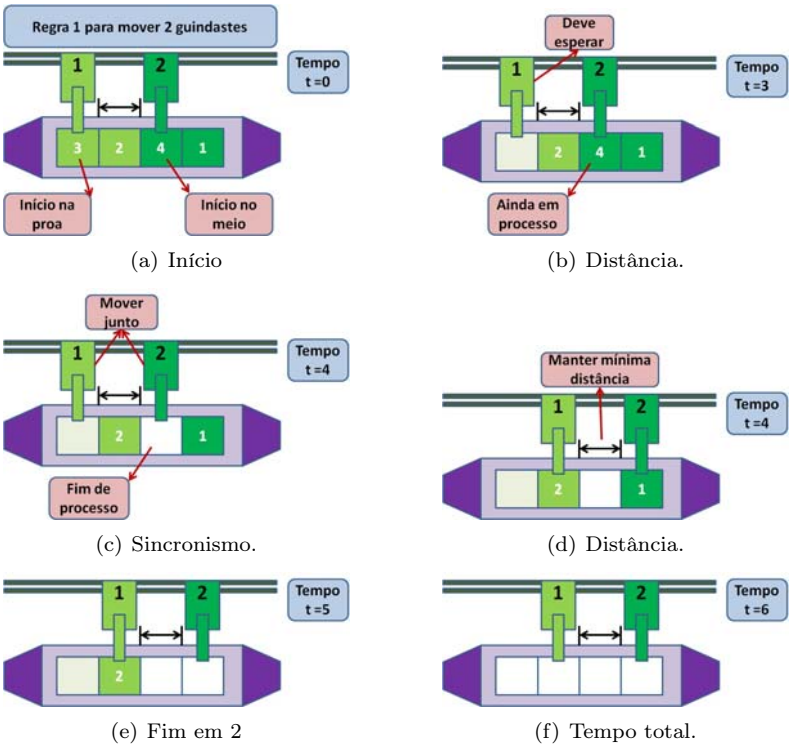
Tabela 8: Regras produzidas pela combinação das regras de carregamento, descarregamento e movimentação de guindastes portuários.

Regra Guindastes	Regra Carregamento	Regra Descarregamento	Regra
Rg1	Rc1	Rd1	1
		Rd2	2
	Rc2	Rd1	3
		Rd2	4
	Rc3	Rd1	5
		Rd2	6
	Rc4	Rd1	7
		Rd2	8
	Rc5	Rd1	9
		Rd2	10
	Rc6	Rd1	11
		Rd2	12
Rg2	Rc1	Rd1	13
		Rd2	14
	Rc2	Rd1	15
		Rd2	16
	Rc3	Rd1	17
		Rd2	18
	Rc4	Rd1	19
		Rd2	20
	Rc5	Rd1	21
		Rd2	22
	Rc6	Rd1	23
		Rd2	24

A diferença entre a codificação apresentada em (Azevedo et al., 2014) e a deste texto é que agora as regras não só definem a disposição dos contêineres no navio, mas, também, como deverá ser realizada a operação dos guindastes portuários, tal como ilustrado na Figura 35.

5.3 Resultados Computacionais para o problema Integrado

Foram obtidos resultados para instância com até cerca de 44 milhões de restrições e 42 milhões de variáveis binárias utilizando um programa criado em Matlab 7.0, e uma máquina com um processador 1.66 GHz Core Duo Intel Processor, memória RAM memory de 2 GB e sistema operacional Windows Vista Operational System com Service Pack 2.Duas funções objetivos foram testadas: uma que busca minimizar o tempo total de operação (Fig. 37) e outra que busca minimizar a instabilidade do arranjo dos contêineres no navio (Fig. 38) para uma instância que considera a movimentação de cargas para um número de portos igual a 30. Sem perda de generalidade foi suposto que o tempo de des-



(a) Início (b) Distância.

(c) Sincronismo. (d) Distância.

(e) Fim em 2 (f) Tempo total.

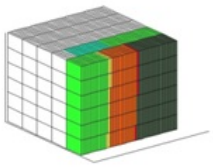
Figura 35: Descrição detalhada do funcionamento da regra 1 de operação de dois guindastes portuários.

P1	4
P2	3
P3	2

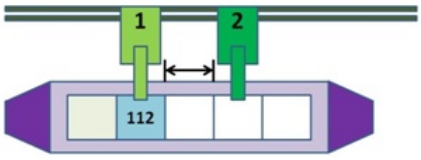
Figura 36: Codificação do algoritmo genético para relacionar indivíduos e soluções.

locamento do guindaste de uma baia para outra e o tempo de movimentação de uma carga seja para carga ou descarga no navio são iguais. Também foi suposto que os pesos e as dimensões de todos os contêineres são iguais. Além disso, tanto o problema de estiva (Fig. 37(a), 37(c); Fig. 38(a), 38(c)) quanto o problema de alocação de guindastes (Fig. 37(b), 37(d); Fig. 38(b), 38(d)) foram resolvidos de forma integrada, isto é, a resolução do problema de estiva (P) observa o impacto da aplicação das regras em termos do problema de alocação de guindastes (O). As soluções mostradas nas Figuras 37 e 38 correspondem aos melhores valores de função objetivo obtidos para a minimização do número de movimentos e a minimização da instabilidade das baias dos navios, respectivamente.

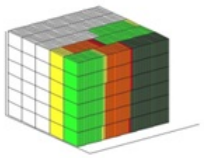
Para o problema com maior número de portos, isto é trinta, foi possível resolver o problema de modo a obter soluções factíveis, bem como melhorar sua qualidade, em menos de 15 minutos, mesmo que o problema resultante tenha dimensão de 44 milhões de restrições e 42 milhões de variáveis apenas para o problema de estiva. Além disso, se no futuro deseja-se integrar outras opera-



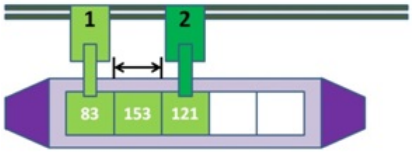
(a) Antes Descarregar Porto 1



(b) *Workload* Porto 1

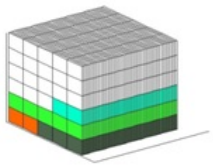


(c) Antes Descarregar Porto 2

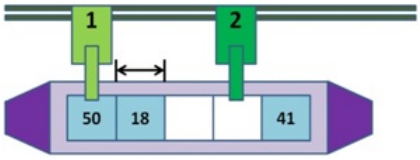


(d) *Workload* Porto 2

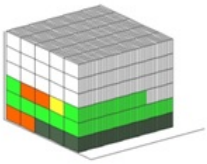
Figura 37: Solução em termos plano de estiva e operação de guindastes para $(\alpha, \beta) = (1, 0)$ na Eq. (2) de modo a minimizar o número de movimentos.



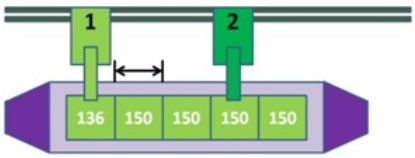
(a) Antes Descarregar Porto 1



(b) *Workload* Porto 1



(c) Antes Descarregar Porto 2



(d) *Workload* Porto 2

Figura 38: Solução em termos plano de estiva e operação de guindastes para $(\alpha, \beta) = (0, 1)$ na Eq. (2) de modo a minimizar a medida de instabilidade.

ções portuárias na metodologia desenvolvida, então, este tempo computacional reduzido pode ser de grande valia para avaliar qual o impacto as operações portuárias podem ter no planejamento de longo prazo fornecido pelo plano de estiva de um dado navio.

6 Conclusões

Este texto procurou fornecer uma primeira primeira visão sobre os problemas de otimização relacionados às operações portuárias. Foi indicada a existência de diversos tipos de portos quanto ao tipo de carga movimentada e particular atenção foi dedicada aos portos com movimentação de carga via contêineres devido a sua importância estratégia de permitir incremento nos fluxos de produtos entre países e assim ter papel central no comércio internacional atual.

Foi relatado o estágio de desenvolvimento atual de efiência, investimento e movimentação de contêineres no Brasil e no mundo. Diante da constatação de que, no Brasil, os custos operacionais são bastante superiores aos praticados em outros em países, e consciente de que o problema é bastante complexo por envolver medidas além dos modelos matemáticos, o presente estudo tentou fornecer ações que podem ser tomadas somente do ponto de vista matemático. Assim, foram apresentados modelos e métodos de solução para três dos cinco principais problemas sobre operações portuárias com vistas a subsidiar soluções que resultem em redução de custos para os agentes envolvidos.

O primeiro problema abordado foi o de alocação de berços. Para este problema foi apresentado um novo método híbrido combinando as meta-heurísticas BRKGA e CS. O BRKGA, isoladamente, não conseguiu bons resultados. Por outro lado, o algoritmo híbrido BRKGA+CS foi capaz de encontrar todas as soluções ótimas, confirmando as características do CS: localização de regiões promissoras e intensificação das buscas. Assim, a combinação entre as meta-heurísticas BRKGA e CS é promissora, pois conforme constatado, é eficiente e robusta na obtenção de soluções viáveis. Além disso, a característica genérica de grande parte dos seus componentes facilita a sua utilização nos mais diversos problemas de otimização sem a necessidade de grandes adequações.

O segundo problema abordado foi o da elaboração do plano de estiva que é um problema NP-Completo de difícil resolução devido às dimensões que assume para instâncias reais. Neste sentido, foi apresentada uma nova representação de soluções para o problema de carregamento de contêineres 3D em terminais portuários (PCCTP 3D). Esta nova representação permite reduzir a quantidade de informações necessárias para se representar uma solução tal que soluções próximas do limitante inferior do número de movimentos podem ser obtidas em pequeno tempo computacional. Na formulação binária do problema do plano de estiva, o número de informações para fornecer uma solução completa é equivalente a um vetor binário de tamanho $(D \times R \times C) \times (N + N^3)$. A representação por regras, aqui utilizada no plano de estiva, o tamanho do vetor é N. Tendo em vista que $D \times R \times C$ expressa o número de contêineres que podem ser armazenados em um navio e que N representa o número de portos, a representação da solução aqui utilizada permite uma enorme redução do tempo computacional para instâncias com grande número de portos. Para tanto, a representação emprega regras que definem como será carregado e descarregado o navio e as regras garantem que as soluções obtidas são sempre factíveis.

A representação por regras também permite a incorporação, no sistema computacional, do conhecimento do profissional responsável por esse serviço no porto. Esta incorporação é feita através da criação e implantação de novas regras. Esta maneira de representar o problema do plano de estiva em conjunto com a aplicação de regras, é uma alternativa promissora na resolução deste problema, até porque assim como se podem acrescentar novas regras, também se

podem utilizar outras heurísticas, como por exemplo, o ILS ou o VNS. Futuramente pretende-se testar uma abordagem multiobjetivo na obtenção de soluções e testes com outras medidas de instabilidade do navio.

O problema de *scheduling* de guindastes também foi tratado, mas de modo que a resolução conjunta do problema de estiva e de alocação de guindastes foram realizadas. Isto é, determinar o arranjo dos contêineres em um navio, mas de modo a observar o correspondente impacto em termos de tempo de operação dos guindastes. Para tanto, a representação por regras foi extendida para ser aplicada não só no problema do plano de estiva, mas também para determinar a programação de guindastes em um terminal portuário de contêineres. A metodologia proposta conseguiu obter soluções para problemas cujo modelo matemático correspondente possui cerca de 44 milhões de restrições e 42 milhões de variáveis em menos de 15 minutos. Diante da velocidade computacional para obter soluções factíveis e de boa qualidade para problemas complexos integrados, uma perspectiva futura é ampliar a utilização da representação por regras de modo a incluir outras operações portuárias tais como a movimentação de contêineres no pátio do porto, bem como considerar diferentes tipos de contêineres em termos de pesos, dimensões e conteúdo (carga tóxica, refrigerada, dentre outros).

7 Agradecimentos

Os autores agradecem à FAPESP pelo apoio financeiro através dos processos 2009/15107-0, 2010/51274-5, 2010-16517-4, 2010/16622-2, 2011/01667-3 e 2014/03258-1.

Referências

A. Imai, K.S.; Nishimura, E. & Papadimitriou, S., Multi-objetive simultaneous stowage and loading planning for a container ship with container rehandle in yard stacks. *European Journal of Operational Research*, 171(2):373–389, 2006.

Aguilera, M.; Roli, A. & Sampels, M., *Hybrid metaheuristics: an emerging approach to optimization.* 1^{st} edição. Berlin/Heidelberg, Alemanha: Springer, 2008.

Al-Khayyal, F. & Hwang, S.J., Inventory constrained maritime routing and scheduling for multi-commodity liquid bulk, part i: Applications and model. *European Journal of Operational Research*, 176(1):106–130, 2007.

Ambrosino, D.; Anghinolfi, D.; Paolucci, M. & Sciomachen, A., An experimental comparison of different heuristics for the master bay plan problem. *Lecture Notes in Computer Science*, 6049:314–325, 2010.

Ambrosino, D.; Sciomachen, A. & Tanfani, E., Stowing a containership: the master bay plan problem. *Transportation Research Part A*, 38(2):81–99, 2004.

Ambrosino, D.; Sciomachen, A. & Tanfani, E., A decomposition heuristics for the container ship stowage problem. *Journal of Heuristics*, 12(3):211–233, 2006.

Amorim, F.M.S., *Metaheurísticas aplicadas ao problema das p-medianas*. Dissertação de mestrado, Centro Federal de Educação Tecnológica de Minas Gerais, Brazil, 2011.

ANTAQ, , Desempenho portuário segundo a agência nacional de transportes aquaviários - antaq. `http://www.antaq.gov.br`, 2013. Accessed: 10-05-2013.

Avriel, M.; Penn, M. & Shpirer, N., Exact and approximate solutions of the container ship stowage problem. *Computers and Industrial Engineering*, 25(1-4):271–274, 1993.

Avriel, M.; Penn, M. & Shpirer, N., Container ship stowage problem: complexity and connection to the coloring of circle graphs. *Discrete Applied Mathematics*, 103(1-3):271–279, 2000.

Avriel, M.; Penn, M. & Wittenboon, S., Stowage planning for container ships to reduce the number of shifts. *Annals of Operations Research*, 76(1):55–71, 1998.

Azevedo, A.; Ribeiro, C.; Chaves, A.; Sena, G.; Leduino, L. & Moretti, A., Solving the 3d container ship stowage problem by using simulated annealing and representation by rules. *In: XXXII Iberian Latin American Congress on Computational Methods in Engineering - CILAMCE 2011*. p. 1–15. 2011a.

Azevedo, A.; Ribeiro, C.; Chaves, A.; Sena, G.; Leduino, L. & Moretti, A., Solving the 3d containership stowage loading planning problem by representation by rules and beam search. *In: First International Conference on Operations Research and Enterprise Systems - ICORES 2012*. p. 132–141. 2012.

Azevedo, A.; Ribeiro, C.; Leduino, L.; Silva, M. & Sivestre, M., Resolução do problema de carregamento e descarregamento 3d de contêineres em terminais portuários via representação por regras e algoritmo genético. *Gestão da Produção, Operações e Sistemas*, 1(4):91–110, 2011b.

Azevedo, A.; Ribeiro, C.; Sena, G.; Chaves, A.; Leduino, L. & Moretti, A., Solving the 3d container ship loading planning problem by representation by rules and meta-heuristics. *International Journal of Data Analysis Techniques and Strategies: Special Issue on Optimisation and Simulation in Realistic Scenarios*, 6(3):228–260, 2014.

Bank, W., Brazil evaluating the macroeconomic and distributional impacts of lowering transportations costs. *In: In: World Bank Other Operational Studies 8083*. p. 1–159. 2008.

Barros, V.H.; Costa, T.S. & A. C.M. Oliveira, L.A.N.L., Model and heuristic for berth allocation in tidal bulk ports with stock level constraints. *Computers & Industrial Engineering*, 60(4):606–613, 2011.

Bausch, D.O.; Brown, G.G. & Ronen, D., Scheduling short-term marine transport of bulk products. *Maritime Policy & Management: The flagship journal of international shipping and port research*, 25(4):335–348, 1998.

Bean, J.C., Genetic algorithms and random keys for sequencing and optimization. *ORSA Journal on Computing*, 6(2):154–160, 1994.

Bernhofen, M.D.; El-Sahli, Z. & Kneller, R., Estimating the effects of the container revolution on world trade. *In: In: CESifo Working Paper Series 4136. CESifo, Center for Economic Studies and Ifo Institute. Munich*. p. 1–33. 2013.

Bierwirth, C. & Meisel, F., A survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, 202(3):615 – 627, 2010.

Botter, R. & Brinati, M., Stowage container planning: a model for getting optimal solution. *In: In: International Conference of Computer Applications in the Automation of Ship-yard Operation and Ship Design*. p. 217–229. 1991.

Braekers, K.; Caris, A. & Janssens, G., Integrated planning of loaded and empty container movements. *OR Spectrum*, 35(2):457–478, 2013.

Buhrkal, K.; Zuglian, S.; Ropke, S.; Larsen, J. & Lusby, R., Models for the discrete berth allocation problem: a computational comparison. *Transportation Research Part E: Logistics and Transportation Review*, 47(4):461–473, 2011.

Buriol, L.S.; Hirsch, M.J.; Pardalos, P.M.; Querido, T.; Resende, M.G.C. & Ritt, M., A biased random-key genetic algorithm for road congestion minimization. *Optimization Letters*, 4(4):619–633, 2010.

Carlo, H.J.; Vis, I.F.A. & Roodbergen, K.J., Storage yard operations in container terminals: Literature overview, trends, and research directions. *European Journal of Operational Research*, 235(2):412–430, 2014.

Cassettari, L.; Mosca, R.; Revetria, R. & Rolando, F., Sizing of a 3,000,000t bulk cargo port through discrete and stochastic simulation integrated with response surface methodology techniques. *International Journal of Systems Applications, Engineering & Development*, 6(1):87–97, 2012.

Chang, D.; Jiang, Z. & W. Yan, J.H., Integrating berth allocation and quay crane assignments. *Transportation Research Part E*, 46(6):975–990, 2010.

Chaves, A.A., *Uma meta-heurística híbrida com busca por agrupamentos aplicada a problemas de otimização combinatória*. Tese de doutorado, INPE, São José dos Campos, Brasil, 2009.

Christiansen, M.; Fagerholt, K. & Ronen, D., Ship routing and scheduling: Status and perspectives. *Transportation Science*, 38(1):1–18, 2004.

Chung, S.H. & Choy, K.L., A modified genetic algorithm for quay crane scheduling operations. *Expert Systems with Applications*, 39(4):4213–4221, 2012.

Cordeau, J.F.; Laporte, G.; Mercier, A. & et al., A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational research society*, 52(8):928–936, 2001.

Cordeau, J.P.; Laporte, G.; Legato, P. & Moccia, L., Models and tabu search heuristics for the berth-allocation problem. *Transportation Science*, 39(4):526–538, 2005.

Costa, T. & de Oliveira, A., Artificial bee clustering search. In: Rojas, I.; Joya, G. & Cabestany, J., (Eds.). Advances in Computational Intelligence. Springer Berlin Heidelberg, v. 7903 de *Lecture Notes in Computer Science*, 2013. p. 20–27.

Croce, F.D. & T'Kindt, V., A recovering beam search algorithm for the one-machine dynamic total completion time scheduling problem. *Journal of the Operational Research Society*, 54(11):1275–1280, 2002.

D. Pacino, R.M.J., Constraint-based local search for container stowage slot planning. *In: In: Proceedings of the International MultiConference of Engineers and Computers Scientists II*. p. 1–6. 2012.

D. Steenken S. Voss, R.S., Container terminal operation and operations research – a classification and literature review. *OR Spectrum*, 26(1):3–49, 2004.

Delgado, A.; Jensen, R.M.; Janstrup, K.; Rose, T.H. & Andersen, K.H., A constraint programming model for fast optimal stowage of container vessel bays. *European Journal of Operational Research*, 220(1):251–261, 2012.

Derrett, C.D.R. & Barrass, B., *Ship Stability for Masters and Mates*. 5a edição. Butterworth-Heinemann, 1999.

Dubrovsky, O.; Levitin, G. & Penn, M., A genetic algorithm with a compact solution encoding for the containership stowage problem. *Journal of Heuristics*, 6:585–599, 2002.

Fan, L.; Low, M.; Ying, H.; Jing, H.; Min, Z. & Aye, W., Stowage planning of large containership with tradeoff between crane workload balance and ship stability. *In: In: Proceedings of the International MultiConference of Engineers and Computers Scientists III*. p. 1–7. 2010.

Fitzgerald, D., *A History of Containerization in the California Maritime Industry: The Case of San Francisco*. PhD thesis, University of California, USA, California Sea Grant College Program, UC San Diego, 1986.

Fox, M.S., *Constraint-Directed Search: A case Study of Job-Shop Scheduling*. PhD thesis, Carnegie-Mellon University, USA, 1983.

Glover, F. & Martí, R., Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 39:653–684, 2000.

Gonçalves, J.F. & Resende, M.G., Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17(5):487–525, 2011.

Guan, Y.; Yang, K.H. & Zhou, Z., The crane scheduling: models and solution approaches. *Annals of Operations Research*, 203(1):119–139, 2013.

Henderson, D.; Jacobson, S. & Johnson, A., *The theory and practice of Simulated Annealing. In: Glover, F.; Kochenberger, G. A. Handbook of Metaheuristics*. 1^{st} edição. New York, USA: Kluwer Academic Publishers, 2003.

Hendriks, M.P.M.; Lefeber, E. & Udding, J.T., Simultaneous berth allocation and yard planning at tactical level. *OR Spectrum*, 35(2):441–456, 2013.

Holland, J., *Adaptation in natural and artificial systems*. 1^{st} edição. USA: The University of Michigan Press, 1975.

Homayouni, S.; Tang, S.; Ismail, N. & Ariffin, M., Using simulated annealing algorithm for optimization of quay cranes and automated guided vehicles scheduling. *International Journal of the Physical Sciences*, 6(27):6286–6294, 2011.

Imai, A.; Chen, H.C.; Nishimura, E. & Papadimitriou, S., The simultaneous berth and quay crane allocation problem. *Transportation Research Part E*, 44(5):900–920, 2008.

Jaramillo, D.I. & Perakis, A.N., Fleet deployment optimization for liner shipping part 2. implementation and results. *Maritime Policy & Management: The flagship journal of international shipping and port research*, 18(4):235–262, 1991.

Javanshir, H. & Ganji, S.S., Yard crane scheduling in port container terminals using genetic algorithm. *Journal of Industrial Engineering International*, 6(11):39–50, 2010.

K. Mak, D.S., Scheduling yard cranes in a container terminal using a new genetic approach. *Engineering Letters*, 17(4):274–280, 2009.

Kim, K.H. & Park, Y.M., A crane scheduling method for port container terminals. *European Journal of Operational Research*, 156(3):752 – 768, 2004.

Kirkpatrick, S.; Gelatt, C. & Vecchi, M., Optimization by simulated annealing. *Journal of the Operational Research Society*, 220(4598):671–680, 1983.

Lee, B.K. & Kim, H., Optimizing the yard layout in container terminals. *OR Spectrum*, 35(2):363–398, 2013.

Lee, D.; Wang, D. & Miao, L., Quay crane scheduling with non-interference constraints in port container terminals. *Transportation Research E*, 44(1):124–135, 2008.

Legato, P.; Mazza, R.M. & Trunfio, R., Simulation-based optimization for the quay crane scheduling problem. *In: WSC '08 Proceedings of the 40th Conference on Winter Simulation*. p. 2717–2725. 2008.

Ligteringen, H. & Velsink, H., *Ports and Terminals*. 1^{st} edição. Delft, The Netherlands: VSSD, 2012.

Maia, H.J., Um choque de concorrência nos portos brasileiros. *Revista Exame*, 1036:48–51, 2013Accessed: 22-05-2013.

Man, K.F.; Tang, K.S. & Kwong, S., *Genetic Algorithms: Concepts and Designs with Disk*. 2a edição. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1999.

Mauri, G.R.; Oliveira, A.C.M.d. & Lorena, L.A.N., Heurística baseada no simulated annealing aplicada ao problema de alocação de berços. *GEPROS-Gestão da Produção, Operações e Sistemas*, 1(1):113–127, 2008.

Meisel, F. & Bierwirth, C., Heuristics for the integration of crane productivity in the berth allocation problem. *Transportation Research Part E: Logistics and Transportation Review*, 45(1):196 – 209, 2009.

Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs*. 3^{rd} edição. Berlin/Heidelberg: Springer-Verlag, 1996.

Mladenovic, N. & Hansen, P., Variable neighborhood search. *Computers and Operations Research*, 24:1097–1100, 1997.

Murty, K.G.; Liu, J.; Wan, Y. & Linn, R., A decision support system for operations in a container terminal. *Decision Support Systems*, 39(3):309–332, 2005.

Ning, Z. & Weijian, M., A model for the stowage planning of 40 feet containers at container terminals. *International Journal of Information Systems for Logistics and Management*, 4(2):41–49, 2009.

Nishimura, E.; Imai, A. & Papadimitriou, S., Berth allocation planning in the public berth system by genetic algorithms. *European Journal of Operational Research*, 131(2):282–292, 2001.

Oceans, T.V., The vast oceans - not too huge for human destruction. `http://tomsviewpoint.blogspot.com.br/2011/11/vast-oceans-not-too-huge-for-human.html`, 2014. Accessed: 01-08-2014.

Oliveira, A.C.A.M.d.; Chaves, A.A. & Lorena, L.A.N., Clustering search. *Pesquisa Operacional*, 33:105 – 121, 2013.

de Oliveira, R.M.; Mauri, G.R. & Nogueira Lorena, L.A., Clustering search for the berth allocation problem. *Expert Systems with Applications*, 39(5):5499–5505, 2012.

Perakis, A.N. & Jaramillo, D.I., Fleet deployment optimization for liner shipping part 1. background, problem formulation and solution approaches. *Maritime Policy & Management: The flagship journal of international shipping and port research*, 18(3):183–200, 1991.

Port, S., Shanghai port, home page. `http://www.portshanghai.com.cn/en/`, 2014. Accessed: 02-08-2014.

Powell, B.J. & Perkins, A.N., Fleet deployment optimization for liner shipping: an integer programming model. *Maritime Policy & Management: The flagship journal of international shipping and port research*, 24(2):183–192, 1997.

R. Stahlbock, S.V., Operations research at container terminals: a literature update. *OR Spectrum*, 30(1):1–52, 2008.

Ribeiro, C. & Azevedo, A., Resolução do problema de carregamento e descarregamento de contêineres em terminais portuários via beam search. *Anais do XLII Simpósio Brasileiro de Pesquisa Operacional*, 1:1–12, 2010.

RNZN, J., Johns rnzn ships through the ages, home page. http://jcsmarinenews.wordpress.com/2013/06/29/only-questions-surround-the-breaking-of-the-mol-comfort/, 2014. Accessed: 01-08-2014.

Sabuncuoglu, I. & Baviz, M., Job shop scheduling with beam search. *European Journal of Operational Research*, 118(2):390–412, 1999.

Sivanandam, S.N. & Deepa, S.N., *Introduction to Genetic Algorithms*. 1a edição. Springer Publishing Company, Incorporated, 2007.

Talbi, E.G., *Metaheuristics: From Design to Implementation*. Wiley Publishing, 2009.

Tavakkoli-Moghaddam, R.; Makui, A.; Salahi, S.; Bazzazi, M. & Taheri, F., Operations research at container terminals: a literature update. *Computers & Industrial Engineering*, 56(1):241–248, 2009.

TCLN, , The cargo law network, home page. http://www.cargolaw.com/2001nightmare_orion.html, 2014. Accessed: 01-08-2014.

Terminal, S.B., Santos brasil terminal, home page. http://www.santosbrasil.com.br/pt-br, 2014. Accessed: 02-08-2014.

Terminals, D.B., Dry bulk terminals, home page. http://drybulkterminals.org/#panel-1, 2014a. Accessed: 02-08-2014.

Terminals, L.B., Ports of gray harbor, home page. http://www.portofgraysharbor.com/terminals/terminal2LiquidBulk.php, 2014b. Accessed: 02-08-2014.

Valente, J. & Alves, R., Filtered and recovering beam search algorithm for the early/tardy scheduling problem with no idle time. *Computers & Industrial Engineering*, 48:363–375, 2005.

Vettorazzo, L., E metade da carga ficou para trás. *Revista Exame*, 1001:60–64, 2011.

Wilson, I. & Roach, P., Container stowage planning: a methodology for generating computerised solutions. *Journal of the Operational Research Society*, 51(11):1248–1255, 2000.

Yang, C.; Wang, X. & Li, Z., An optimization approach for coupling problem of berth allocation problem and quay crane assignment in container terminal. *Computers & Industrial Engineering*, 63(1):243–253, 2012.

Zeng, Q. & Yang, Z., Integrating simulation and optimization to schedule loading operations in container terminals. *Computers & Operations Research*, 36(6):1935–1944, 2009.

A Apêndice

A.1 Codificando PCCTP 2D em AMPL

A formulação de programação linear inteira do PCCTP 2D, proposta por Avriel et al. (1998), é dada pelas Eqs. (24)-(29).

$$\min \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \sum_{v=i+1}^{j-1} \sum_{r=1}^{R} \sum_{c=1}^{C} x_{ijv}(r,c) \tag{24}$$

subject to

$$\sum_{v=i+1}^{j} \sum_{r=1}^{R} \sum_{c=1}^{C} x_{ijv}(r,c) - \sum_{k=1}^{i-1} \sum_{r=1}^{R} \sum_{c=1}^{C} x_{kij}(r,c) = T_{ij}$$
$$i = 1, \cdots, N-1; j = i+1, \cdots, N \tag{25}$$
$$\sum_{k=1}^{i} \sum_{j=i+1}^{N} \sum_{v=i+1}^{j} x_{kjv}(r,c) = y_i(r,c)$$
$$i = 1, \cdots, N-1; r = 1, \cdots, R; c = 1, \cdots, C \tag{26}$$
$$y_i(r,c) - y_i(r+1,c) \geq 0$$
$$i = 1, \cdots, N-1; r = 1, \cdots, R; c = 1, \cdots, C \tag{27}$$
$$\sum_{i=1}^{j-1} \sum_{p=j}^{N} x_{ipj}(r,c) + \sum_{i=1}^{j-1} \sum_{p=j+1}^{N} \sum_{v=j+1}^{p} x_{ipv}(r+1,c) \leq 1 \tag{28}$$
$$x_{ijv}(r,c) = 0 ou 1$$
$$i = 1, \cdots, N; r = 1, \cdots, R; c = 1, \cdots, C. \tag{29}$$

onde: a variável binária $x_{ijv}(r,c)$ é definida de forma que assume o valor 1 se existir um contêiner no compartimento (r,c) que foi ocupado no porto i e tem como destino final o porto j e movido no porto v; caso contrário assume valor zero. Por compartimento (r,c) entende-se a linha r, a coluna c, e uma única baia para todo compartimento de carga do navio. Similarmente, a variável $y_i(r,c)$ possui valor 1 se saindo do porto i o compartimento (r,c) for ocupado por um contêiner; caso contrário assume valor 0. A restrição (25) é a restrição de conservação de fluxo, onde T_{ij} é o elemento da matriz de transporte que representa o número de contêineres que embarcam no porto i com destino ao porto j. A restrição (26) em cada compartimento (r,c) será ocupado por no máximo um contêiner. A restrição (27) é necessária para garantir que existem contêineres embaixo do contêiner que ocupa o compartimento (r,c). A restrição (28) é responsável por definir a movimentação dos contêineres: se um contêiner que ocupa a posição (r,c) é descarregado no porto j, então, ou não existem contêineres acima dele, ou o índice v do contêiner que ocupa o compartimento $(r+1,c)$ não é maior que j.

A função objetivo da Eq. (24) possui é tal que minimiza a movimentação dos contêineres para todos os portos. O custo total de movimentação dos contêineres assume que a movimentação de um contêiner possui um custo unitário e igual para todos os portos.

É possível codificar em AMPL o modelo matemático do PCCTP 2D tal como apresentado a seguir.

```
/* Containership problem: only movements objective function */

  # Parameters to define the sets.
  param first > 0 integer;              # first port index.
  param last > first integer;           # last port index.
  param ri > 0 integer;                 # first row index.
  param rf > ri integer;                # last  row index.
  param ci > 0 integer;                 # first column index.
  param cf > ci integer;                # last  column index.
  param di > 0 integer;                 # first bay index.
  param df >= di integer;               # last  bay index.


  # Defining the sets.
  set N 'portos' := first..last;        # set of all ports.
  set Nu:= first..last-1;               # set of 1..ports - 1.
  set Nl:= first+1..last;               # set of 2..ports.
  set R := ri..rf;                      # set of all ship rows.
  set Ru:= ri..rf-1;                    # set of ship rows - 1.
  set Rl:= ri+1..rf;                    # set of ship rows >=2.
  set C := ci..cf;                      # set of all ship columns.
  set D := di..df;                      # set of all ship bays.

  #Parameters to store the model results in a file.
  param FileResults, symbolic, default "ContainershipResults.txt";

  /* transportation matrix */
  param T{i in N, j in N}, >= 0;

  /* container movements variables */
  var x{i in N, j in N, v in N, r in R, c in C, d in D} binary;

  /* containership spaces variables */
  var y{i in N, r in R, c in C, d in D} binary;

  /* Only the number of movements */
  minimize z: sum{i in Nu} sum{j in N: j >= i+1} sum{v in N: v >= (i+1)
and v <= (j-1)} sum{r in R} sum{c in C} sum{d in D} x[i,j,v,r,c,d];

  /* Container conservation */
  s.t. Shipquant{i in Nu, j in N: j >= i+1}:  (sum{v in N: v >= i+1
and v <= j} sum{r in R} sum{c in C} sum{d in D}  x[i,j,v,r,c,d]) -
(sum{k in N: k <= i-1} sum{r in R} sum{c in C} sum{d in D} x[k,j,i,r,c,d])
= T[i,j];

  /* At most one container  */
  Slot{i in Nu, r in R, c in C, d in D}: (sum{k in N: k <= i} sum{j
in N: j >= i+1} sum{v in N: v >= i+1 and v <= j} x[k,j,v,r,c,d]) =
y[i,r,c,d];
```

```
  /* No floating containers allowed */
  OnTop{i in Nu, r in Ru, c in C, d in D}:  y[i,r,c,d] - y[i,r+1,c,d]
>= 0;

  /* Rules for moving container on top of a specific stack*/
  Shift{j in Nl, r in Ru, c in C, d in D}: (sum{i in N: i <= j-1} sum{p
in N: p >= j} x[i,p,j,r,c,d]) + (sum{i in N: i <= j-1} sum{p in N:
p >= j+1} sum{v in N: v >= j+1 and v <= p} x[i,p,v,r+1,c,d]) <= 1;

  solve;

  /* REPORT */
  printf '\n'
  >> FileResults;
  printf '-------------------------------------------------------\n'
  >> FileResults;
  printf 'Solution Found \n'
  >> FileResults;
  printf '\n'
  >> FileResults;
  for{i in N}
  {
    printf '-------------------------------------------------------\n'
    >> FileResults;
    for{j in N}
    {
      for{v in N}
      {
        printf '----------------------\n'
        >> FileResults;
        printf ' (i = %5d, j = %5d, v = %d) \n', i, j, v
        >> FileResults;
        printf '----------------------\n'
        >> FileResults;
        printf '----------------------\n'
        >> FileResults;
        printf ' X[r,c,d] \n'
        >> FileResults;
        printf '----------------------\n'
        >> FileResults;
        for{d in D}
          for{r in R}
          {
            for{c in C}
            {
             /*printf{r in R, c in C, d in D} "  %5s =  %5g  \n ",
j, x[i,j,v,r,c,d]*/
             printf "  %2d   ", x[i,j,v,r,c,d]
             >> FileResults;
```

```
            }
            printf '  \n   '
            >> FileResults;
          }
        printf '----------------------\n'
        >> FileResults;
      }
    }
 }
 for{i in N}
 {
    printf '-------------------------------------------------------\n'
    >> FileResults;
    printf ' (i = %5d) \n', i
    >> FileResults;
    printf '-------------------------------------------------------\n'
    >> FileResults;
    for{d in D}
      for{r in R}
      {
        for{c in C}
        {
           /*printf{r in R, c in C, d in D} "  %5s =  %5g  \n ", j,
x[i,j,v,r,c,d]*/
          printf "  %2d   ", y[i,r,c,d]
          >> FileResults;
        }
        printf '  \n   '
        >> FileResults;
      }
      printf '----------------------\n'
      >> FileResults;
 }

 printf 'Total (z): %10.2f\n', (sum{i in Nu} sum{j in N: j >= i+1}
sum{v in N: v >= (i+1) and v <= (j-1)} sum{r in R} sum{c in C} sum{d
in D} x[i,j,v,r,c,d]);
 /*>> FileResults;*/
 printf '----------------------\n'
 >> FileResults;
 printf '\n'
 >> FileResults;

 /*-------------------------------------------------------------------------------------*/
 /* SEÇÃO DE DADOS
                  */
 /*-------------------------------------------------------------------------------------*/

 data;
```

```
/*
param first := 1;
param last  := 6;
param ri    := 1;
param rf    := 4;
param ci    := 1;
param cf    := 5;
param di    := 1;
param df    := 1;
*/
param first := 1;
param last  := 5;
param ri    := 1;
param rf    := 4;
param ci    := 1;
param cf    := 4;
param di    := 1;
param df    := 1;


/*:       Port1  Port2  Port3  Port4  Port5 :=*/
param T :  1      2      3      4      5:=
1          0      2      5      0      0
2          0      0      2      3      1
3          0      0      0      2      2
4          0      0      0      0      1
5          0      0      0      0      0;

/*:       Port1  Port2  Port3  Port4  Port5 :=*/
/*param T :  1      2      3      4      5     6:=
1          0      9      2      5      3     0
2          0      0      4      1      3     0
3          0      0      0      0      3     4
4          0      0      0      0      2     2
5          0      0      0      0      0     4;*/


end;
```

A.2 Elaborando as regras de carregamento e descarregamento

A regra de carregamento Re1 tem seu código em Matlab dado como se segue.

```
%-------------------------------------------------------------%
%-------------------------------------------------------------%
% Elaboracao da Regra R1 de Entrada.                          %
%-------------------------------------------------------------%
%-------------------------------------------------------------%
% Procedimento de preenchimento:                              %
%-------------------------------------------------------------%
%  (1) Considera-se que a ordem de preenchimento
%      das cargas no navios ->, ou seja, da
%      esquerda para a direita de baixo para cima.
%      m ^
%        |
%        |
%      1 ------>
%        1     n
%
%  (2) Recebe uma matriz M vazia ou parcialmente
%      preenchida, bem como um vetor D com cargas
%      do i-esimo porto.
%
%  (3) Retorna a matriz M com as cargas contidas no
%      vetor D preenchidas de acordo com a regra RE1.
%-------------------------------------------------------------%
function [vM] = tdfre1(vM,D)


  % Ordenando os elementos de D em ordem decrescente
  % de porto.
  [qq, ind] = sort(D,'descend');

  % Laco para preencher a matriz usando ->.
  d = length(vM);
  t=1;
  k=1;
  % Numero de conteineres a serem embarcados.
  tam = length(D);

  % Laco para buscar espacos vazios para passar
  % containeres do vetor D para a matriz D. A busca
  % de espacos vazios comeca nas linhas inferiores
  % percorrendo as colunas de cada linha vai preenchendo
  % a matriz M e vai da camada da proa ate a popa.
  while ((t <= d)&&(k <= tam))

    % Extraindo a matriz M de vM.
```

```
46      M = vM{t};
47
48      % Obtendo as dimensoes de uma dada camada.
49      [m,n]=size(M);
50
51      % Iniciando o laco de preenchimento da i-esima camada
            .
52      i=1;
53      while ((i <= m)&&(k <= tam))
54        j=1;
55        while ((j <=n)&&(k <= tam))
56          % Se houver um espaco vazio, entao, ponha um
                container
57          % que esta em v.
58          if (M(i,j) == 0)
59            M(i,j) = D(ind(k));
60            k = k + 1;
61          end
62          j = j + 1;
63        end
64        i = i + 1;
65      end
66
67      % Reobtendo a matriz M preenchida para vM.
68      vM{t} = M;
69
70      t = t + 1;
71    end % Fim do laco para percorrer todas as camadas.
72
73    % Mensagem de aviso de infactibilidade do problema !!
        Pois,
74    % nao houve espaco suficiente para colocar todos os
        conteineres.
75    if (k < tam)
76      fprintf(1,'Problema na funcao tdfre1 !! \n');
77      fprintf(1,'Nao foi possivel carregar todos os
          containeres!! \n');
78      fprintf(1,'Espaco insufiente !! \n');
79    end
80
81
82  %-------------------------------------------------------%
```

A regra de descarregamento Rs1 tem seu código em Matlab dado como se segue.

```
1  %-------------------------------------------------------%
2  %-------------------------------------------------------%
3  % Elaboracao da Regra R1 de Saida.                      %
4  %-------------------------------------------------------%
5  %-------------------------------------------------------%
```

```
6  % Procedimento de preenchimento:                                %
7  %-------------------------------------------------------------%
8  %  (1) Procede-se a retirada das cargas verificando-se
9  %      para cada coluna quantas movimentos sao
       necessarios
10 %      para se retirar as cargas do porto k.
11 %
12 %      m ^
13 %        |
14 %        |
15 %      1 ------>
16 %        1     n
17 %
18 %  (2) Ao se retirar uma carga de uma coluna
       imediatamente
19 %      a carga que se encontra acima eh movida para baixo
       .
20 %      Se a carga acima for tambem uma carga cujo destino
21 %      eh o porto k, entao, passa-se a uma carga mais
       acima,
22 %      ate se chegar a ultima carga ! Se nao existir
       nenhuma
23 %      carga acima, indica-se que existem espacos vazios
       na
24 %      coluna.
25 %
26 %  (3) Retorna a matriz M com as cargas contidas no
27 %      vetor D retiradas de acordo com a regra RS1 e o
       numero
28 %      de movimentos que foram necessarios.
29 %-------------------------------------------------------------%
30 function [vM,nmov,vC] = tdfrs1(vM,k)
31
32   % Zerando o numero de movimentos realizados.
33   nmov = 0;
34   % Inicializando o vetor com conteineres a serem
35   % descarregados agora para permitirem o movimento
36   % de containeres a serem descarregados no porto
37   % atual. Para serem carregados novamente no navio
38   % estes conteineres utilizarao uma regra de
         carregamento.
39   vC = [];
40
41   % Obtendo o numero de baias do navio.
42   d = length(vM);
43
44   % Laco para percorrer as camadas do navio: da proa
45   % ate a popa.
46   for t=1:d
47
```

```
48      % Extraindo a matriz M de vM.
49      M = vM{t};
50
51      % Capturando a dimensao da matriz M.
52      [m,n]=size(M);
53
54      % Lacos para percorrer M e retirar as cargas
55      % do navio que devem ser descarregadas no
56      % porto k.
57
58      % Varrendo cada coluna e retirando todos os
59      % containeres k da mesma.
60      for j=1:n
61
62        % Percorrendo todas as linhas de uma dada
63        % coluna j fixada.
64        i=1;
65        % Ver se o elemento em foco eh o container
66        % que deve ser descarregado.
67        while ((i <= m)&&(M(i,j) ~= k))
68          i = i + 1;
69        end
70
71        % Se encontrou o container do tipo k na coluna.
72        if (i <= m)
73          % Contabilizar movimentos de descarga.
74          nmov = nmov + 1;
75          % Indice do local onde foi encontrado o
76          % container a ser removido.
77          t1 = i;
78          % Indice do local onde foi encontrado o
79          % container que pode ser realocado para
80          % o espaco onde foi removido um container.
81          t2 = i+1;
82
83          % Laco que percorre toda a coluna ate que
84          % nao existam mais containeres a serem
85          % descarregados e consequemente movidos.
86          while ((t2 <= m)&&(M(t2,j) ~= 0))
87            % Verificar se o proximo container da
88            % coluna sera retirado ou se podera
89            % ocupar o espaco deixado por outro
90            % container a ser descarregado.
91            % Nao eh container a ser retirado,
92            % portanto deve ser deslocado.
93            if (M(t2,j) ~= k)
94              % Este container devera ser retirado e depois
95              % colocado novamente no navio, mas segundo
                    uma
96              % dada regra de entrada. Portanto, seu
```

```
97                  % movimento conta apenas 1 vez.
98                  nmov = nmov + 1;
99                  vC(length(vC)+1) = M(t2,j);
100                 % O container eh k, ou seja, deve ser
                        retirado e
101                 % portanto so devemos contabilizar o
                        movimento
102                 % de retirada do mesmo.
103               else
104                 nmov = nmov + 1;
105               end
106               t2 = t2 + 1; % Analisar containeres mais acima.
107             end % Fim while.
108
109           % Containeres que foram retirados devem ser
                  apagados.
110           for tt=t1:m
111             M(tt,j) = 0;
112           end
113
114         end  % Fim do if.
115
116       end     % Fim for de j.
117
118       % Reobtendo a matriz M preenchida para vM.
119       vM{t} = M;
120
121       t = t + 1;
122     end % Fim do laco para percorrer todas as camadas.
123
124
125 %-------------------------------------------------------------------------%
```

Notas Biográficas

Anibal Tavares de Azevedo tem mestrado, doutorado e pós-doutorado em Engenharia Elétrica (UNICAMP, 2002, 2006 e 2007, respectivamente). Foi Professor Assistente da UNESP-Guaratinguetá (2007-2011) e atualmente é Professor Assistente da Faculdade de Ciências Aplicadas da UNICAMP. Tem interesse na área de metaheurísticas e otimização combinatória, como aplicações em problemas de sistemas de potência, planejamento e controle da produção, telecomunicações e logística portuária.

Antônio Augusto Chaves tem doutorado e pós-doutorado em Computação Aplicada (INPE, 2009). Foi Professor Assistente da UNESP-Guaratinguetá (2010) e atualmente é Professor Assistente da UNIFESP-São José dos Campos. Seus interesses são computação, aprendizado para pesquisa operacional, problemas de otimização combinatória, formulações matemáticas, heurísticas e metaheurísticas.

Luiz Leduino de Salles Neto tem mestrado e doutorado em Matemática Aplicada (UNICAMP, 2000 e 2005, respectivamente). Realizou pós-doutorado na universidade de Sevilla em 2010. Atualmente é Professor Assistente da UNIFESP-São José dos Campos. Seus interesses são pesquisa operacional, problemas de corte e empacotamento, problemas combinatórios, não-lineares e multiobjetivo.