ALGORITMO HÍBRIDO COM CS E BRKGA APLICADO AO PROBLEMA DE ALOCAÇÃO DE BERÇOS

Carlos Zaca Pomari
1Universidade Federal de São Paulo – UNIFESP, Instituto de Ciência e Tecnologia
Rua Talin, 330, CEP 12231-280, São José dos Campos, SP, Brasil
cpomari@sabesp.com.br

Antonio Augusto Chaves
1Universidade Federal de São Paulo – UNIFESP, Instituto de Ciência e Tecnologia
Rua Talin, 330, CEP 12231-280, São José dos Campos, SP, Brasil
antonio.chaves@unifesp.br

RESUMO

Este trabalho aborda o problema de alocação de berços (PAB), que busca minimizar o tempo de atendimento dos navios nos portos. Esse problema possui forte interesse teórico e prático, devido ao crescimento da economia mundial e ao comércio internacional de bens que têm estimulado, recentemente, a demanda por serviços de transporte marítimo. A metodologia consiste no estudo e aplicação de um algoritmo híbrido baseado nas meta-heurísticas *Biased Random Key Genetic Algorithm* (BRKGA) e, *Clustering Search* (CS) para resolver de forma heurística o PAB. Inicialmente, o BRKGA é aplicado de forma isolada e depois como gerador de soluções para o processo de agrupamento do CS. Os testes computacionais com instâncias disponíveis na literatura validaram o método, pois ele foi capaz de encontrar todas as soluções ótimas. Paralelamente, verificou-se um benefício mútuo: o BRKGA simplificando o processo de agrupamento do CS e este provendo eficiência e robustez na obtenção de soluções viáveis.

PALAVARAS CHAVE. Alocação de Berços, Biased Random Key Genetic Algorithm, Clustering Search.

Área principal (Meta-heurísticas)

ABSTRACT

This work addresses the berth allocation problem (BAP), which seeks to minimize the vessels' handling times in ports. This problem has a strong theoretical and practical interest, since the growth of the world economy and international trade in goods has stimulated in recent years, the demand for shipping services. The methodology for the development of this work consists in the study and application of a hybrid algorithm based on Biased Random Key Genetic Algorithm (BRKGA) and Clustering Search (CS) metaheuristics to solve approximately the BAP. This study uses the BRKGA alone and then as solutions generator to the CS´s clustering process. The computational tests with instances available in the literature validated the method, since it was able to find all optimal solutions. Furthermore, we observed a mutual benefit: The BRKGA simplifying the CS' clustering process and, this providing efficiency and robustness in obtaining feasible solutions.

KEYWORDS. Berth Allocation, Biased Random Key Genetic Algorithm, Clustering Search.

Main area (Metaheuristics)

1. Introdução

O crescimento da economia mundial e do comércio internacional de mercadorias incentivou, nos últimos anos, a procura de serviços de transporte marítimo, o que pode ser observado pelo aumento do número de contêineres transportados desde o início da década: cerca de 150% conforme apresentado por UNCTAD (2009). Nessa mesma direção, segundo a CCE (2009), o número de navios em operação deverá ter um acréscimo estimado em 29% até 2018.

No Brasil, é possível verificar um crescimento expressivo na movimentação de cargas em instalações portuárias e em terminais de uso privativo (TUPs): no levantamento realizado pela ANTAQ (2012), em 1990 o acumulado de movimentação de cargas nessas instalações foi de 360 milhões de toneladas. Dez anos depois, esse montante alcançou a cifra de 834 milhões, ou seja, um crescimento da ordem de 130%.

É importante destacar que, segundo esse mesmo levantamento, o crescimento tem ocorrido independente do tipo de carga transportada e o aumento da capacidade dos navios e o volume de carga manejada nos portos e nos TUPs acompanham essa tendência.

Entretanto, segundo Hijjar e Alexim (2006), os investimentos não seguem esse crescimento, desencadeando gargalos de acesso aos terminais intermodais, tanto portuários quanto ferroviários. Devido a isso, o país não consegue aproveitar de forma ótima essa situação de desenvolvimento, pois não garante condições necessárias para o escoamento da produção.

Com foco nesse pensamento, uma solução seria o investimento na construção de novas instalações de atraque e em adaptar as já existentes para receber navios mais largos e profundos, isto é, investindo de forma a acomodar uma maior variedade de tipos de embarcações. Por outro lado, adequações de carácter estrutural, por si só, não garantem que o atendimento seja eficiente, pois a gestão de um complexo portuário envolve também uma diversidade de problemas de tomada de decisão, que segundo Silva (2008), podem ocorrer nos níveis: estratégico, tático e operacional.

Um dos problemas operacionais que merece destaque é o plano de atracação, no qual objetiva-se combinar o berço mais adequado para atender cada navio, de modo a respeitar as restrições impostas e minimizar o tempo de espera e atendimento dos mesmos. Como a elaboração desse plano envolve a manipulação de diversas informações, muitas delas dinâmicas, Mauri *et al.* (2008) sugerem o uso da computação com auxílio de técnicas de otimização visando, através da simulação de diferentes cenários, agilizar a tomada de decisão.

Devido a essa busca por uma logística em acomodar e minimizar o tempo de espera e atendimento dos navios, surgiram inúmeros problemas na literatura. Dentre eles, o Problema de Alocação de Berços (PAB) que consiste em determinar a ordem de atracação dos navios e seus respectivos berços. Além do interesse prático, o PAB pelo fato de ser um problema *NP - difícil* (Cordeau *et al*., 2005), mostra-se atraente tanto no estudo de novas modelagens quanto na aplicação de heurísticas para sua resolução.

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) e *Biased Random Key Genetic Algorithm* (BRKGA). 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.

O restante do artigo está organizado como segue. A Seção 2 apresenta uma breve revisão bibliográfica sobre o PAB. Na Seção 3 uma formulação matemática para o PAB é descrita. Este trabalho baseia-se numa relaxação dessa formulação e, a mesma pode ser examinada na subseção 3.1. A Seção 4 contém um resumo dos conceitos das meta-heurísticas CS

e BRKGA, além da mescla de ambas (BRKGA+CS) e, os resultados computacionais obtidos são apresentados na Seção 5. Por fim, as conclusões são resumidas na Seção 6.

2. Revisão bibliográfica

O PAB é um problema que possui forte interesse teórico e prático, desse modo, procura-se ilustrar, resumidamente, o que foi encontrado na literatura.

Para Cordeau *et al.* (2005) o objetivo é a minimização (ponderada) do tempo de serviço total para todos os navios, definido como o tempo decorrido entre a chegada no porto, atracação e o tempo de serviço. Os autores formulam o problema como Roteamento de Veículos com Múltiplas Garagens e Janelas de Tempo. Duas versões do PAB são consideradas: o caso discreto e o caso contínuo, uma vez que o caso discreto nem sempre satisfaz as restrições espaciais.

Com base na formulação apresentada por Cordeau *et al.* (2005), Mauri *et al.* (2008) aplicaram relaxações e penalizações à mesma, chegando a um modelo capaz de representar um Problema de Roteamento de Veículos com Garagens Múltiplas, sem Janelas de Tempos. A intenção era desenvolver um modelo cuja resolução fosse menos complexa. Uma vez que consideraram um conjunto finito de berços e ignoraram suas dimensões espaciais, o mesmo foi tratado em sua forma discreta. Baseando-se sua heurística no *Simulated Annealing* (SA) desenvolveram três tipos de movimentos de troca para a geração de novas soluções vizinhas. Concluíram que o método proposto se mostrou eficiente visto os pequenos *gaps* apresentados, indicando grande proximidade com as soluções ótimas para o problema.

Buhrkal *et al.* (2009), motivados pelo PAB ser considerado um dos mais importantes problemas para qualquer terminal de contêineres, revisaram e explanaram três modelos principais para o PAB do tipo dinâmico e discreto. O primeiro, proposto por Imai *et al.* (2001), o segundo apresentado em Cordeau *et al.* (2005) e por fim o discutido por Christensen e Holst (2008). Esses modelos foram comparados a partir dos seus resultados computacionais e, concluíram que o PAB modelado como um *generalized set-partitioning problem* (GSPP) de Christensen e Holst (2008) superou todos os outros. Nesse modelo, o tempo é discretizado e, uma restrição garante que, a qualquer momento, somente um navio pode estar num determinado berço num intervalo de tempo específico. Uma coluna no modelo representa um navio em uma determinada posição de atracação e num determinado intervalo de tempo. Os autores ainda confirmam que o PAB modelado como um GSPP encontrou, pela primeira vez, todos os valores ótimos para as instâncias propostas por Cordeau *et al.* (2005).

Oliveira *et al.* (2009), visando melhorar os resultados obtidos por Mauri *et al.* (2008), desenvolveram um algoritmo (CS+SA) baseado na meta-heurística *Clustering Search* (CS) e utilizaram o SA proposto por Mauri *et al.* (2008) como gerador de soluções para o CS. Obtiveram as soluções ótimas (ratificadas por Buhrkal *et al.*, 2009) em todos os casos e com redução do tempo computacional. Desse modo, evidencia-se a eficiência e robustez do CS na localização de regiões promissoras e como alternativa para acelerar a obtenção de boas soluções.

3. Formulação matemática para o PAB

Nesse trabalho o PAB é tratado de forma semelhante ao apresentado por Cordeau *et al.* (2005), Mauri *et al.* (2008) e Oliveira *et al.* (2009), 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 (ANTAQ, 2012) 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 1 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.

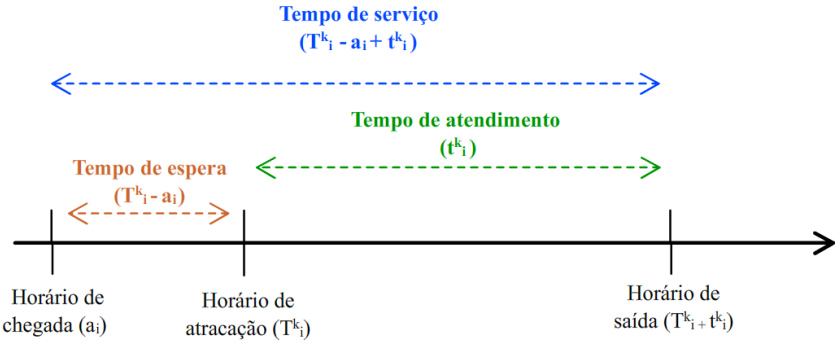

Figura 1: Cronograma de atendimento de um navio - Cordeau *et al.* (2005)

Com isso, Cordeau *et al.* (2005) modelaram o PAB como um Problema de Roteamento de Veículos com Múltiplas Garagens e Janelas de Tempo (PRVGMJT), considerando os navios como clientes e os berços, garagens, de modo que a cada garagem é alocado um veículo, num total de *m* veículos. Ainda, considera-se que cada veículo inicia e termina seu trajeto em sua garagem. Desse modo, esses navios são modelados como vértices em um multi-grafo. Cada garagem é dividida em um vértice de origem *o(k)* e um de destino *d(k)*. As janelas de tempo, nos vértices *o(k)* e *d(k)*, correspondem ao período de funcionamento dos berços. O PRVGMJT é especificado como um multi-grafo $G^k = (V^k, A^k)\ \forall k \in M$, onde $V^k = N \cup \{o(k), d(k)\}$ e $A^k \subseteq V^k \times V^k$ (Oliveira *et al.*, 2009).

Assim, pode-se definir matematicamente o PAB por meio da seguinte notação: N: conjunto de navios, $n = |N|$; M: conjunto de berços, $m = |M|$; 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 término da janela de tempo para o navio i; v_i: valor do tempo de serviço do navio i; $x_{ij}^k = 1$ se o navio j é atendido pelo berço k após o navio i, para $x_{ij}^k \in \{0,1\} \forall k \in M$, $\forall (i, j) \in A^k$; T_i^k é o horário que o navio i atracou no berço k, $\forall k \in M$, $i \in N$; $T_{o(k)}^k$ é o horário em que o primeiro navio atracou no berço k, $\forall k \in M$; $T_{d(k)}^k$ é o horário em que o último navio saiu do berço k, $\forall k \in M$; e $M_{ij}^k = \max(b_i + t_i^k - a_i, 0)$, $\forall k \in M$, $\forall (i, j) \in N$.

$$\min \quad Z = \sum_{i \in N} \sum_{k \in M} v_i \left(T_i^k - a_i + t_i^k \sum_{j \in N \cup \{d(k)\}} x_{ij}^k \right) \tag{1}$$

Sujeito à:

$$\sum_{k \in M} \sum_{j \in N \cup \{d(k)\}} x_{ij}^k = 1 \quad \forall i \in N \tag{2}$$

$$\sum_{j \in N \cup \{d(k)\}} x_{o(k),j}^k = 1 \quad \forall k \in M \tag{3}$$

$$\sum_{i \in N \cup \{o(k)\}} x_{i,d(k)}^k = 1 \quad \forall k \in M \tag{4}$$

$$\sum_{j \in N \cup \{d(k)\}} x_{ij}^k - \sum_{j \in N \cup \{o(k)\}} x_{ji}^k = 0 \quad \forall k \in M, \forall i \in N \tag{5}$$

$$T_i^k + t_i^k - T_j^k \leq (1 - x_{ij}^k)\ M_{ij}^k \quad \forall k \in M, \forall (i,j) \in A^k \tag{6}$$

$$T_i^k \geq a_i \quad \forall k \in M, \quad \forall i \in N \tag{7}$$

$$T_i^k + t_i^k \sum_{j \in N \cup \{d(k)\}} x_{ij}^k \leq b_i \quad \forall k \in M, \forall i \in N \tag{8}$$

$$T^k_{o(k)} \geq s^k \qquad \forall k \in M \tag{9}$$

$$T^k_{d(k)} \leq e^k \qquad \forall k \in M \tag{10}$$

$$x^k_{ij} \in \{0,1\} \forall k \in M \, , \, \forall (i,j) \in A^k \tag{11}$$

A função objetivo (1) minimiza a soma do tempo de serviço, associado a um custo operacional; as restrições (2) são responsáveis em garantir que cada navio é atendido por apenas um berço; as restrições (3) e (4) são necessárias para assegurar que para cada berço *k*, um navio será o primeiro a ser atendido e outro será o último; as restrições (5) garantem a "conservação do fluxo", ou seja, o atendimento para os demais navios; as restrições (6) impedem que um navio atraque num berço antes do navio anterior liberá-lo. Nelas são considerados apenas os navios válidos para cada berço, ou seja, alguns navios não podem ser atendidos em determinados berços devido a restrições técnicas, por exemplo, calados, profundidades e equipamentos disponíveis no berço; as restrições (7) e (8) impedem que o horário de atração de um navio seja inferior ao seu horário de chegada e nem que seu horário final de atendimento seja superior ao horário-limite do navio (janela de tempo); as restrições (9) e (10) operam de forma semelhante às restrições (7) e (8), no entanto, são relacionadas aos berços, isto é, um berço não pode ser utilizado se estiver indisponível (tempo de disponibilidade do berço); por fim, as restrições (11) são empregadas para garantir que o domínio das variáveis de decisão seja binário, isto é, para que elas assumam apenas os valores 0 ou 1.

3.1. Relaxação da formulação matemática

Mauri *et al.* (2008) sugerem a relaxação das restrições (7), (8), (9) e (10), de forma que o problema passa a se comportar como o Problema de Roteamento de Veículos com Garagens Múltiplas, sem Janelas de Tempos. Segundo os autores, isso torna menos complexa a resolução do problema, pois elimina as janelas de tempo. Assim, as restrições (7) e (8) são transferidas para o termo (13) da função objetivo, e as restrições (9) e (10) são inseridas no termo (14). Ressalta-se que a relaxação do problema não o impede de apresentar as mesmas soluções do problema original (com janelas de tempo), por outro lado, também não pode impedir soluções inviáveis. Para evitá-las, eliminando-as de forma transparente durante a execução do algoritmo, são inseridos os coeficientes de penalização ($\omega = [\omega_0, \omega_1, \omega_2]$) em cada termo da função. Sendo assim, o tempo de serviço, juntamente com seu custo, é representado na expressão (12). A expressão (13) é responsável por minimizar violações nas janelas de tempo dos navios, enquanto que a expressão (14) faz o mesmo para os berços.

Dessa forma, tem-se a seguinte formulação:

$$\min \quad Z^* = \omega_0 \sum_{i \in N} \sum_{k \in M} v_i \left(T^k_i - a_i + t^k_i \sum_{j \in N \cup \{d(k)\}} x^k_{ij} \right) + \tag{12}$$

$$\omega_1 \sum_{i \in N} \sum_{k \in M} \left(\max\left(0, a_i - T^k_i\right) + \max\left(0, T^k_i + t^k_i \sum_{j \in N \cup \{d(k)\}} x^k_{ij} - b_i \right) \right) + \tag{13}$$

$$\omega_2 \sum_{k \in M} \left(\max\left(0, s^k - T^k_{o(k)}\right) + \max\left(0, T^k_{d(k)} - e^k\right) \right) \tag{14}$$

Sujeito à (2) – (6) e (11).

4. Algoritmo híbrido BRKGA+CS

Neste trabalho o método proposto por Gonçalves e Resende (2010, 2011), o chamado Algoritmo Genético de Chaves Aleatórias Viciadas (BRKGA, do inglês *Biased Random Key*

Genetic Algorithm), foi aplicado ao Problema de Alocação de Berços (PAB), inicialmente isoladamente e depois como gerador de soluções para o processo de agrupamento do *Clustering Search* (CS), proposto por Chaves (2009).

4.1 BRKGA

O BRKGA é uma meta-heurística recente e baseada no Algoritmo Genético de Chaves Aleatórias (*Random Keys Genetic Algorithm* (RKGA), em inglês) proposto em 1993 por Bean (1993) para resolver problemas de sequenciamento (Gonçalves e Resende, 2011), tais como m*ultiple 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, como por exemplo em: Mendes *et al.* (2009), Buriol *et al.* (2009), entre outros. 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 e 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 e 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, 1993 e Gonçalves e 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 2.

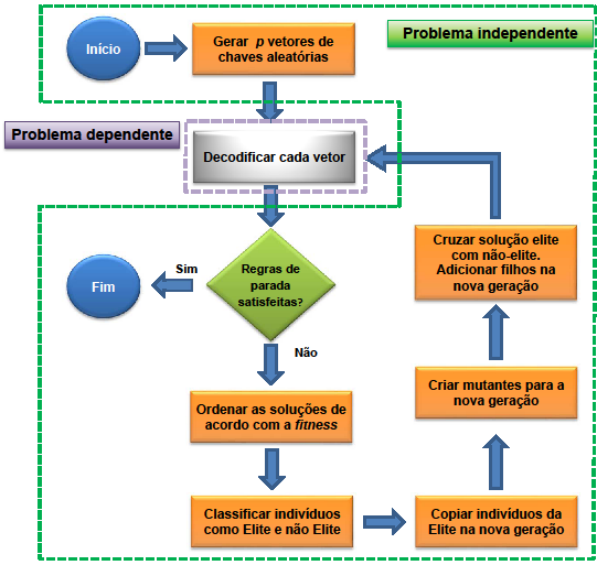

Figura 2: Fluxograma do BRKGA - Gonçalves e Resende (2011)

4.2 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. Chaves (2009), 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.*, 2009).

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 *et al.* (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 lo-

cal realiza uma intensificação da busca no seu centro.

A Figura 3 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.

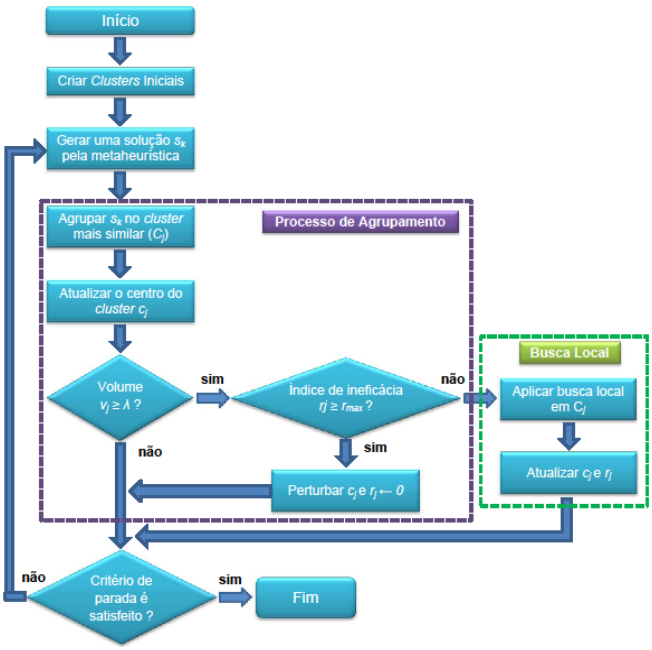

Figura 3: Fluxograma do método CS – Chaves (2009)

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 r_{max}. 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.

4.3 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 4.

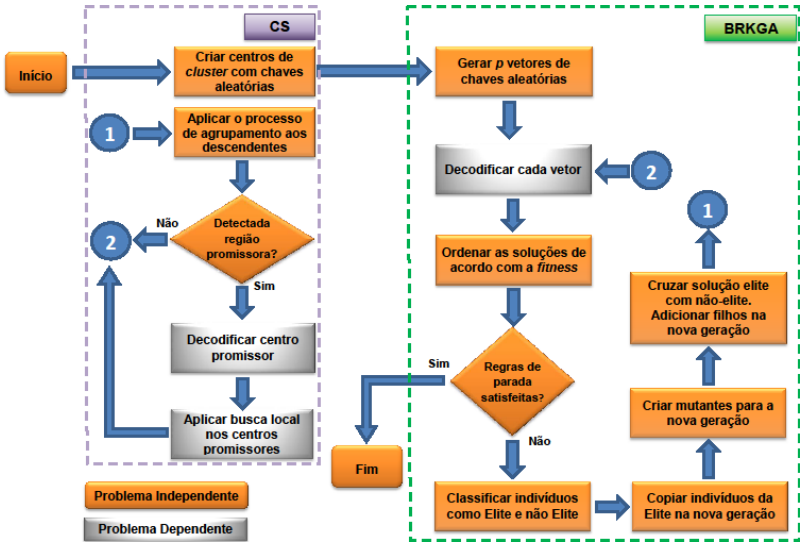

Figura 4: Fluxograma BRKGA+CS

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 *Berço1* enquanto que]0,5;1], o *Berço2*. 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: *Berço1*: *navios 1* e *5*; *Berço2*: *navios 4*, *3* e 2.

O processo de assimilação ficou a cargo do método Reconexão por Caminhos (PR, do inglês *Path-Relinking*) (Glover *et al.*, 2010). 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 (15), cujo valor (*de*) é calculado diretamente nas chaves aleatórias.

$$de = \sqrt{\sum_{i=1}^{n} (ch2_i - ch1_i)^2} \qquad (15)$$

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: [ω0,ω1,ω2] = [1,10,10], assim como as buscas locais definidas pelos movimentos Reordenar, 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; p_e = 0,20; p_m = 0,20 e ρ_e = 0,65. E, para o BRKGA+CS: p = 200; p_e = 0,25; p_m = 0,15 e ρ_e = 0,65. Os do CS foram: NC = 20, λ = 4 e r_{max} = 300.

A estratégia de refinamento das soluções no processo de otimização local, foi a *Variable Neighborhood Descent* (VND) onde as heurísticas citadas anteriormente são executadas conforme descrito, de forma simplificada, pelo pseudocódigo da Figura 5. 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*.

```
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 5: Pseudocódigo VNS

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

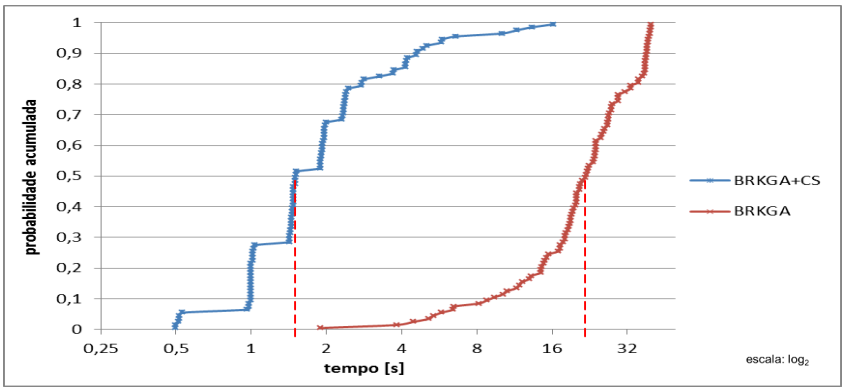

Figura 6: *Time to target*

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.* (2005) 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.* (2009) executado num PC *Intel Xeon* de 2.66 GHz e por fim o CS de Oliveira *et al.* (2009) 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.* (2005) 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.

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
Média		120,00		60,26		14,98		12,79		20,55		38,39	**0,00**	0,88

(1) Cordeau *et al.* (2005) - (2) Mauri *et al.* (2008) - (3) Buhrkal *et al.* (2009) - (4) Oliveira *et al.* (2009)

Tabela 1: Comparação dos resultados com outros métodos da literatura

6. Conclusões

No caso deste trabalho, a implementação do CS foi simplificada com a utilização do BRKGA como meta-heurística geradora de soluções, pois partes importantes do método se tornaram genéricas podendo ser aproveitadas noutros problemas de otimização, uma vez que passaram a trabalhar com vetores de chaves aleatórias e não com soluções do problema. Dentre elas, destacam-se os processos de agrupamento, de assimilação e perturbação.

O BRKGA, isoladamente, não conseguiu bons resultados. Esse mesmo fato foi apontado por Amorim (2011), que estudou a aplicação do BRKGA na resolução do Problema das *p*-Medianas não capacitado. A autora só obteve melhores resultados associando a busca local FSB (do inglês *Fast Swap-Based*) ao BRKGA. 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. 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. Sugere-se, como trabalhos futuros, um aperfeiçoamento dessa mescla, de forma a desenvolver um *framework* nos moldes daquele apresentado por Toso e Resende (2012), no caso do BRKGA.

7. Referências

Amorim, F. M. S. (2011), Meta-heurísticas aplicadas ao problema das *p*-medianas. *Dissertação* (Mestrado em Modelagem Matemática e Computacional).
ANTAQ (2012), Anuário estatístico aquaviário, 2012. URL http://www.antaq.gov.br/, 2013.
Bean, J. C. (1993), Genetics and random keys for sequencing e optimization. *Technical report*, 92, 1993.
Buhrkal, K., Zuglian, S., Ropke, S., Larsen, J. e Lusby, R. (2009), Models for the discrete berth allocation problem: a computational comparison. *Technical Report*, 14:1–24, 2009.
Buriol, L. S., Hirsch, M. J., Pardalos, P. M., Querio, T., Resende, M. G. C. e Ritt, M. (2009), A hybrid genetic algorithm for road congestion minimization**.** *Anais do XLI Simpósio Brasileiro de Pesquisa Operacional (SBPO 2009)*, Porto Seguro, Brasil.
Buriol, L. S., Hirsch, M. J., Pardalos, P. M., Querio, T., Resende, M. G. C. e Ritt, M. (2010), A biased random-key genetic algorithm for road congestion minimization. *Optimization Letters*, 4,619–633.
CCE (2009), Objetivos estratégicos e recomendações para a política comunitária de transporte marítimo no horizonte de 2018. *Comunicação da Comissão ao Parlamento Europeu*, 1–15 (http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=COM:2009:0008:FIN:PT:PDF, 2013).
Chaves, A. A. (2009), Meta-heurísticas híbridas com busca por agrupamentos para problemas de otimização combinatória. *Tese* (Doutorado em Computação Aplicada).
Christensen, **C. G. e Holst C. T.** (2008), Berth allocation in container terminals. *Dissertação* (Mestrado em Matemática) - Universidade da Dinamarca.
Cordeau, J. F., Laporte, G. e Mercier, A. (2001), A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational Research Society*, 52, 928-936.
Costa, M. F., Fiedler, N. C. e Mauri, G. R. (2013), Clustering search e simulated annealing para resolução do problema de escalonamento de motoristas no transporte de madeira. *Scientia Forestalis*, 41(99):299–305.
Glover, F., Laguna, M. e Martí, R. (2010), Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 29,653–684.
Gonçalves, J. F e Resende, M. G. C. (2010), A parallel multipopulation biased random-key genetic algorithm for a container loading problem. *AT&T Labs Research Technical Report*.
Gonçalves, J. F e Resende, M. G. C. (2011), Biased random key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17,487–525.
Hijjar, M. F. e Alexim, F. M. B. (2006), Avaliação do acesso aos terminais portuários e ferroviários de contêineres no brasil. *Instituto COPPEAD de Administração* – UFRJ.
Imai, A., Nishimura, E. e Papadimitrious, S. (2001), The dynamic berth allocation problem for a container port. *Transportation Research Part B: Methodological*, 35(4):401–417.
Mauri, G. R., Oliveira, A. C. M. e Lorena, L. A. N. (2008), Heurística baseada no simulated annealing aplicada ao problema de alocação de berços, *GEPROS - Gestão da Produção, Operações e Sistemas*, 1(1), 113-127.
Mendes, J. J. M., Gonçalves, J. F. e Resende, M. G. C. (2009), A random key based genetic algorithm for the resource constrained project scheduling problem. *Computers & Operations Research*, 36(1):92–109.
Oliveira, A. C. M., Chaves A. A. e Lorena, L. A. N. (2013), Clustering search. *Pesquisa Operacional*, 33, 105–121.
Oliveira, R. M., Mauri, G. R. e Lorena, L. A. N. (2009), Clustering search aplicado ao problema de alocação de berços. *XLII Simpósio Brasileiro de Pesquisa Operacional*, 1651–1662.
Silva, V. M. D. (2008), Um modelo heurístico para alocação de navios em berços. *Dissertação*(Mestrado em Engenharia) – Universidade Federal de Santa Catarina.
Toso, R. F. e Resende, M. G. C. (2012), A c + + application programming interface for biased random-key genetic algorithms. *AT&T Labs Research Technical report*.
UNCTAD (2009), Review of maritime transport. United Nations Publication, (E.09.II.D.11):219. URL http://unctad.org/en/p./PublicationArchive.aspx?publicationid=1696