20 e 21 de outubro
Instituto Nacional de Pesquisas Espaciais - INPE
São José dos Campos - SP

Abordagem da Metaheurística *Clustering Search* com *Simulated Annealing* para o Problema de Alocação de Berços de Navios

Rudinei Martins de Oliveira[1,*], Geraldo Regis Mauri[2], Luiz Antonio Nogueira Lorena[3]

[1]rudmart@gmail.com, INPE, Brasil

[2]mauri@cca.ufes.br, UFES, Brasil

[3]lorena@lac.inpe.br, INPE, Brasil

Resumo. *Este trabalho apresenta uma revisão da literatura e algumas abordagens para o Problema de Alocação de Berços (PAB) de navios em portos. Devido à crescente demanda de navios que transportam containeres, o PAB pode ser considerado como um dos principais problemas em terminais marítimos. Nesse contexto, é proposta uma nova alternativa para resolvê-lo. Essa alternativa é baseada na aplicação do método Clustering Search (CS) com a metaheurística Simulated Annealing (SA). O CS é um método iterativo que divide o espaço de busca em clusters e é composto por uma metaheurística, um processo de agrupamento e uma heurística de busca local. Além disso, o CS utiliza como entrada as soluções correntes geradas pelo SA. Por fim, os resultados obtidos são comparados a métodos recentes encontrados na literatura, permitindo assim verificar sua eficiência.*

Palavras-chave: *Alocação de Berços, Clustering Search, Simulated Annealing.*

1. Introdução

Em 2008 a frota de navios que transportam containeres teve um aumento em sua capacidade de 17,3 milhões de toneladas, ou 11,9%, e passaram a representar 13,6% do total mundial. No início de 2009, a frota mercante mundial atingiu 1,19 milhões, um crescimento de 6,7% em comparação a janeiro de 2008 e, desde o início da década, a quantidade de containeres aumentou em 154% (UNCTAD, 2009).

Desse modo, devido ao intenso fluxo de navios e containeres nos portos, estes são forçados a investir pesadamente para acomodar os navios, aprofundando e alargando

canais e construindo novas instalações de atraque, tudo para que o tempo de atendimento do navio seja o menor possível.

Assim, a busca por uma logística de acomodar e minimizar o tempo de espera e atendimento dos navios motivou o surgimento de um problema conhecido na literatura como Problema de Alocação de Berços (PAB). O PAB consiste em alocar navios a posições de atraque de forma que seja utilizado o máximo de espaço do cais minimizando o tempo de serviço. As decisões a serem tomadas dizem respeito à posição e ao tempo em que o navio deverá atracar (Imai *et al*., 2001).

O PAB possui grande quantidade de restrições físicas, técnicas, entre outras. Isso faz com que seja possível modelá-lo de diferentes maneiras. Quanto aos aspectos espaciais dos berços, o PAB pode ser modelado como discreto, contínuo ou híbrido (Imai *et al*., 2005). No caso discreto, o cais é dividido em vários berços e somente um navio é atendido de cada vez em cada berço, independente do seu tamanho. No caso contínuo, não há nenhuma divisão do cais e, dessa forma, os navios podem atracar em qualquer posição. Já no caso híbrido, como no caso discreto, o cais é dividido em berços, mas os navios grandes podem ocupar mais de uma posição, permitindo assim que navios pequenos compartilhem seu berço. Além disso, se for levado em conta a chegada dos navios, o problema pode ser tratado como estático ou dinâmico (Imai *et al*., 2001). O caso estático assume que todos os navios já estão no porto para o atendimento dinâmico, o caso dinâmico permite aos navios chegarem a qualquer momento.

Em ambos os casos, busca-se por uma melhor distribuição do espaço minimizando o tempo total de permanência dos navios no porto. Nesse contexto, este trabalho apresenta uma nova alternativa para resolver o PAB. É proposta uma aplicação do método híbrido conhecido como *Clustering Search* - CS (Chaves, 2009), utilizando o *Simulated Annealing* como gerador de soluções. O CS proposto é comparado a métodos recentes encontrados na literatura, permitindo assim verificar sua eficiência na resolução do PAB.

O restante do artigo está organizado como segue. A Seção 2 apresenta uma breve revisão bibliográfica sobre o PAB. Na Seção 3 é apresentada uma formulação matemática existente, e uma relaxação dessa formulação, que é utilizada como base neste trabalho, é descrita na Seção 4. A Seção 5 apresenta de forma detalhada o CS proposto, e os resultados computacionais obtidos são apresentados na Seção 6. Por fim, as considerações finais são resumidas na Seção 7.

2. Revisão bibliográfica

Os trabalhos iniciais acerca do PAB surgiram no final dos anos 80, quando Thurman (1989) propôs um modelo de otimização para o planejamento de berços para a estação naval Norfolk (EUA). A partir desse modelo, Brown *et al*. (1994) elaboraram um plano para minimizar os conflitos dos carregamentos nessa mesma estação. Os mesmos autores ainda apresentaram um planejamento de berços para submarinos em Brown *et al*. (1997). São raros os trabalhos relacionados ao PAB até meados dos anos 90. Entretanto, tais trabalhos vêm ganhando foco, principalmente nos últimos 10 anos.

Imai *et al*. (2001) abordaram o PAB em sua forma dinâmica. Os autores apresentam um método baseado na relaxação lagrangiana do problema original. Dois anos depois, Imai

et al. (2003) aprimoraram sua abordagem considerando diferentes prioridades de atendimento entre os navios. Além disso, os autores propuseram um Algoritmo Genético como método de solução.

Cordeau *et al*. (2005) propõem duas formulações e duas heurísticas baseadas na Busca Tabu para resolver o PAB. Os autores apresentam testes realizados para o porto de Gioia Tauro (Itália). Cheong *et al*. (2008) apresentam uma aplicação do método Multiobjective Evolutionary Algorithm (MOEA) para resolver o PAB. Giallombardo *et al*. (2010) apresentam um modelo de programação quadrática e um de programação linear para representar o PAB. Além disso, os autores utilizam uma Busca Tabu e uma técnica de programação matemática para resolver instâncias baseadas em dados reais.

No Brasil, Mauri *et al*. (2008a) propõem uma abordagem baseada na aplicação do *Simulated Annealing* para resolução do caso discreto do PAB. Os autores tratam o problema como um Problema de Roteamento de Veículos com Múltiplas Garagens e Janelas de Tempo (PRVGMJT). Os resultados computacionais superam os obtidos pelo CPLEX e pela Busca Tabu proposta por Cordeau *et al*. (2005). Por fim, Mauri *et al*. (2008b) tratam o PAB com um método híbrido chamado ATP/PL, que utiliza o Algoritmo de Treinamento Populacional em conjunto com um modelo de Programação Linear por meio da técnica de Geração de Colunas. Os resultados obtidos superam os apresentados em Mauri *et al*. (2008a).

3. Formulação matemática para o PAB

Considerando o PAB em sua forma discreta e dinâmica, assim como descrita por Cordeau *et al*. (2005), o PAB pode ser modelado como um PRVGMJT (Cordeau *et al*., 2001; Mauri *et al*., 2008a), onde os navios são vistos como clientes e os berços como garagens. Dessa forma, existem m veículos, um para cada garagem. Cada veículo inicia e termina seu tour em sua garagem. Esses navios são modelados como vértices em um multi-grafo. Cada garagem é dividida em um vértice de origem e um de destino. Nos vértices inicial e final, as janelas de tempo 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 $A^k \subseteq V^k$ x V^k. Assim, utilizou-se a seguinte notação: N: conjunto de navios, $n = |N|$; M: conjunto de berços, $m=|M|$; t^k_i: 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; $x^k_{ij} \in \{0,1\}, \forall k \in M,\ \forall (i,j) \in A^k$, $x^k_{ij} =$ 1 se o navio j é atendido pelo berço k após o navio i; $T^k_i\ \forall k \in M$, $i \in N$ é o horário em que o navio i atracou no berço k; $T^k_{o(k)}\ \forall k \in M$ é o horário em que o primeiro navio atracou no berço k; $T^k_{d(k)}\ \forall k \in M$ é o horário em que o último navio saiu do berço k; $M^k_{ij} = \max\{b_i + t^k_i - a_j, 0\},\ \forall k \in M,\ \forall (i,j) \in N$. A Figura 1 ilustra os intervalos de tempo utilizado por cada navio, além das diferentes variáveis utilizadas na formulação do PAB. Em seguida, é apresentada a formulação matemática para o PAB proposta por Cordeau *et al.* (2005).

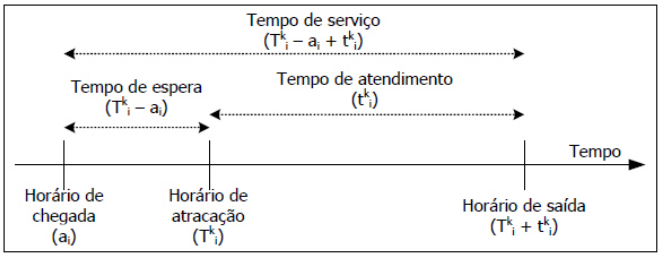

Figura 1. A Representação das variáveis de tempo (Mauri *et al*., 2008a).

Minimizar:

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

Sujeito a:

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

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

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

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

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

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

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

$$T_{o(k)}^k \ge s^k \qquad \forall k \in M \tag{9}$$

$$T_{d(k)}^k \le e^k \qquad \forall k \in M \tag{10}$$

$$x_{i,j}^k \in \{0,1\} \qquad \forall k \in M, \forall (i,j) \in A^k \tag{11}$$

A função objetivo (1) minimiza a soma do tempo de serviço, ponderada por um custo associado v_i. A restrição (2) indica que cada navio é atendido por apenas um único berço. As restrições (3) e (4) asseguram que para cada berço k um navio será o primeiro e outro será o último a ser atendido. A restrição (5) garante a conservação do fluxo de atendimento para os navios restantes. A restrição (6) indica a consistência do horário de atracação dos navios. As restrições (7) e (8) garantem que o horário de atracação seja maior que o tempo de chegada e o horário de saída do navio seja menor que seu tempo limite de atendimento (janela de tempo). As restrições (9) e (10) garantem o tempo de disponibilidade do berço. Por fim, a restrição (11) define o domínio das variáveis de decisão.

4. Formulação relaxada para o PAB

Mauri *et al*. (2008a) propõem a relaxação das restrições (7), (8), (9) e (10), de tal forma que, 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). Além disso, coeficientes de penalização

($\omega = [\omega_0, \omega_1, \omega_2]$) são adicionados em cada termo da função. Dessa forma, tem-se a seguinte formulação:

Minimizar:

$$\omega_0 \sum_{i \in N} \sum_{k \in M} v_i \left(T_i^k - a_i + t_i^k \sum_{j \in N \cup \{d(k)\}} x_{ij}^k \right) + \tag{12}$$

$$\omega_1 \sum_{i \in N} \sum_{k \in M} \left(\max\left(0, a_i - T_i^k\right) + \left(\max\left(0, T_i^k + t_i^k \sum_{j \in N \cup \{d(k)\}} x_{ij}^k - b_i \right) \right) \right) + \tag{13}$$

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

Sujeito a:

$$\sum_{k \in M} \sum_{j \in N \cup \{d(k)\}} x_{ij}^k = 1 \qquad \forall i \in N \tag{15}$$

$$\sum_{j \in N \cup \{d(k)\}} x_{o(k)j}^k = 1 \qquad \forall k \in M \tag{16}$$

$$\sum_{i \in N \cup \{o(k)\}} x_{i,d(k)}^k = 1 \qquad \forall k \in M \tag{17}$$

$$\sum_{j \in N \cup \{d(k)\}} x_{i,j}^k - \sum_{j \in N \cup \{o(k)\}} x_{j,i}^k = 0 \qquad \forall k \in M, \forall i \in N \tag{18}$$

$$T_i^k + t_i^k - T_j^k \leq (1 - x_{i,j}^k) M_{i,j}^k \qquad \forall k \in M, \forall (i, j) \in A^k \tag{19}$$

$$x_{i,j}^k \in \{0,1\} \qquad \forall k \in M, \forall (i, j) \in A^k \tag{20}$$

A partir dessa formulação, tem-se uma nova função objetivo (12), (13) e (14). Em cada termo dessa função há um fator de penalidade indicados pelos ômegas. O termo (12) mantém na função objetivo o tempo de serviço junto com um custo associado. No terno (13) as violações das janelas de tempo dos navios são minimizadas. Por último, o termo (14) minimiza as violações nas janelas de tempo dos berços. Segundo Mauri *et al*. (2008a) com base nessa nova formulação do PAB e avaliando suas restrições, nota-se que apesar do problema ter sido modificado para um problema menos árduo (Problema de Roteamento de Veículos com Garagens Múltiplas sem Janelas de Tempo) ele poderá apresentar as mesmas soluções do problema original (com janelas de tempo). Por outro lado, o modelo também poderá apresentar soluções inviáveis, mas essas inviabilidades são eliminadas por meio das penalizações inseridas no modelo.

5. CS proposto

Segundo Chaves (2009) o CS é um método iterativo que procura dividir o espaço de busca e localizar regiões promissoras por meio do enquadramento dessas em clusters. Um cluster pode ser definido por três atributos $C = \{c,v,r\}$. O centro c_i é uma solução que representa o *cluster i*, e identifica a sua localização dentro do espaço de busca. O volume v_i é a quantidade de soluções agrupadas no *cluster i*. Um *cluster* se torna promissor quando o volume atingir um certo limitante λ. O índice de ineficácia r_i é uma variável de controle para identificar se a busca local está ou não melhorando o centro do *cluster i*. O valor de r_i indica o número de vezes consecutivas que a busca local foi aplicada no *cluster i* e não melhorou a solução. Esse atributo evita que a busca local seja executada em regiões ruins ou regiões que já tenham sido suficientemente exploradas por mais de r_{max} vezes.

O CS é formado basicamente por três componentes principais: uma metaheurística geradora de soluções, um processo de agrupamento e uma heurística de busca local. A cada iteração do CS, uma solução S é gerada pela metaheurística e enviada para o processo de agrupamento. Essa solução é então agrupada no *cluster* mais similar C_j e o centro desse *cluster* c_i é atualizado com informações contidas na nova solução agrupada, fazendo com que o centro se desloque no espaço de busca.

Em seguida, é analisado o volume v_j do *cluster* e, caso esse volume atinja um limitante λ ($v_i \geq \lambda$), percebe-se que algum padrão de solução está sendo predominantemente gerado pela metaheurística. Portanto, esse *cluster* pode estar em uma região de busca promissora. Por fim, é analisado o índice de ineficácia r_j, ou seja, caso a heurística de busca local não melhore a solução por r_{max} vezes consecutivas ($r_j \geq r_{max}$), é aplicada uma perturbação aleatória no centro c_j, objetivando escapar de um possível ótimo local. Por outro lado, se $r_j < r_{max}$, a heurística de busca local é aplicada no centro c_j analisando a vizinhança do *cluster*. Encerrado esse processo, retorna-se para a metaheurística que irá gerar uma nova solução. O critério de parada do CS é geralmente definido pela metaheurística escolhida. A Figura 2 apresenta o fluxograma de execução do CS. Mais detalhes sobre esse método são apresentados em Chaves (2009) e Oliveira (2004).

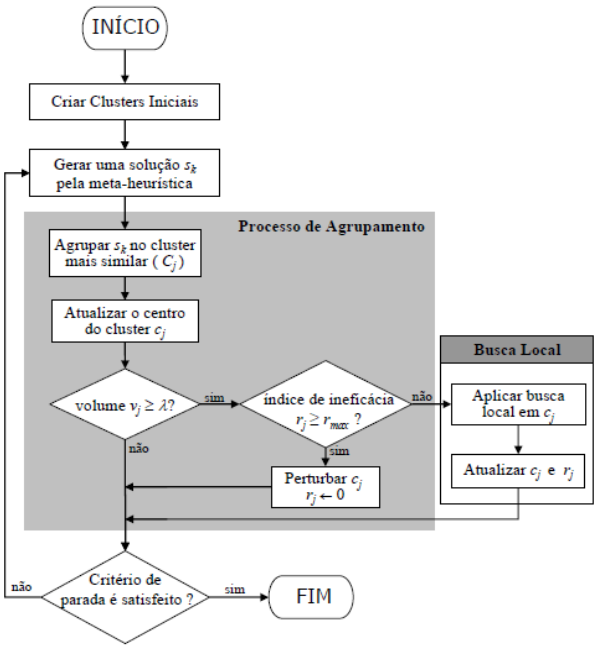

Figura 2. Fluxograma do CS (Chaves, 2009).

Seguindo o fluxograma do CS (Figura 2), são criados então os clusters iniciais. Assim, para cada cluster é criada uma solução por meio das heurísticas de distribuição e programação apresentadas em Mauri *et al*. (2008a). Essas heurísticas são apresentadas nas Figuras 3 e 4, respectivamente.

1. CRIAR (*m* berços vazios);
2. CRIAR (uma lista *L* com todos os navios);
3. ORDENAR (a lista *L* pelo horário de chegada dos navios ao porto);
4. PARA (cada navio *j* em *L*, *j* = 1,2,...,*n*) FAÇA
5. SELECIONAR (um berço *i*, *i* = 1,2,...,*m*);
6. SE (o berço *i* não puder atender ao navio *j*)
7. VOLTAR (para o passo 5);
8. SENÃO
9. ATRIBUIR (o navio *j* ao berço *i*);
10. FIM-SE;
11. FIM-PARA;

Figura 3. Heurística de distribuição (Mauri *et al*., 2008a).

A partir de então, o SA, baseado no proposto por Mauri *et al*. (2008a), é executado, e a cada temperatura, a solução corrente (não a melhor) é enviada ao CS. A Figura 5 apresenta um pseudo-código do SA implementado. Pode-se notar que o CS é chamado na linha 20 desse algoritmo, ou seja, a cada temperatura.

1. PARA (cada berço *k*, *k* = 1,2,...,*m*) FAÇA
2. PARA (cada navio *i* atribuído a *k*) FAÇA
3. $$T_i^k = \begin{cases} \max(a_i, s^k), & i = 1 \\ \max(a_i, T_{i-1}^k + t^k{}_{i-1}), & i > 1 \end{cases}$$
4. FIM-PARA;
5. FIM-PARA;
6. CALCULAR (a função objetivo para a solução atual);

Figura 4. Heurística de programação (Mauri *et al*., 2008a).

```
1.  DADO (α, SAmax, T0 e TC) FAÇA
2.  GERAR (uma solução S por meio da heurística de distribuição);
3.  AVALIAR (a solução S por meio da heurística de programação);
4.  S* ← S;                    {Melhor solução obtida até então}
5.  IterT ← 0;                 {Número de iterações na temperatura T}
6.  T ← T0;                    {Temperatura corrente}
7.  ENQUANTO (T > TC) FAÇA
8.     ENQUANTO (IterT < SAmax) FAÇA
9.        IterT ← IterT + 1;
10.       GERAR (um vizinho qualquer S' por meio de um dos mov. de troca);
11.       APLICAR (a heurística de programação em todos os berços de S');
12.       Δ ← f(S') – f(S);
13.       SE (Δ < 0)   S ← S';
14.          SE (f(S') < f(S*))    S* ← S';   FIM-SE;
15.       SENÃO
16.          TOMAR (x ∈ [0,1]);
17.          SE (x < e^(-Δ/T))    S ← S';   FIM-SE;
18.       FIM-SE;
19.    FIM-ENQUANTO;
20.    EXECUTAR-CS (solução corrente S);
21.    T ← α * T;    IterT ← 0;
22. FIM-ENQUANTO;
23. S ← S*;
24. RETORNAR (S);
```

Figura 5. Algoritmo *Simulated Annealing* utilizado no CS (Adaptado de Mauri *et al*., 2008a).

Como estrutura de vizinhança no SA (linha 10) foram utilizados três diferentes movimentos de troca: *Reordenar navios*, *Realocar navio* e *Trocar navios* (Mauri *et al*., 2008a). Assim como na geração da solução inicial, esses movimentos garantem que cada navio seja atribuído apenas a berços que possam atendê-los.

Após a execução de cada um dos movimentos de troca, a heurística de programação é aplicada para eliminar as sobreposições e recalcular o valor da função objetivo da nova solução. Esses movimentos são apresentados em Mauri *et al*. (2008a,b).

Cada solução vizinha no SA (linha 10) é gerada por apenas um desses movimentos, sendo a sua escolha feita de forma aleatória, porém uniformemente distribuída, possibilitando assim uma boa diversidade entre as soluções intermediárias geradas, e consequentemente uma boa exploração do espaço de soluções.

O CS é implementado de acordo com o apresentado em Chaves (2009). Vale destacar que antes mesmo da execução do SA, as soluções "centro" de clusters já foram criadas (como descrito anteriormente). Ao final da execução do SA, ou seja, do método CS-SA como um todo, a melhor solução encontrada é tomada como solução final para o problema. O algoritmo EXECUTAR-CS é apresentado na Figura 6.

```
1.  DADA (uma solução S);
2.  ENCONTRAR (o cluster C_j mais similar a S);
3.  v_j ← v_j + 1;
4.  ATUALIZAR (o centro do cluster C_j);
5.  SE v_j ≥ λ ENTÃO
6.      v_j ← 1;
7.      SE r_j ≥ r_max ENTÃO
8.          APLICAR (perturbação em c_j)
9.          r_j ← 0;
10.     SENÃO
11.         APLICAR (busca local – encontrar c_j');
12.         SE f(c_j') < f(c_j) ENTÃO
13.             c_j ← c_j';
14.             r_j ← 0;
15.         SENÃO
16.             r_j ← r_j + 1;
17.         FIM-SE;
18.     FIM-SE;
19. FIM-SE;
```

Figura 6. Algoritmo EXECUTAR-CS.

Como mencionado anteriormente, a determinação do cluster mais similar (linha 2) é dada pela menor distância de Hamming (Hamming, 1950). A atualização do centro do cluster (linha 4) é dada pela execução do *Path-Relinking* entre a solução dada S e a solução c_j centro do cluster C_j (Figura 7).

A idéia desse algoritmo é simples, e consiste em executar os movimentos necessários para "transformar" a solução S' (cópia de S) na solução c_j. A partir destes movimentos, a melhor solução encontrada é tomada como novo centro do cluster C_j.

```
1.  DADO (S' e c_j)
2.  PARA (cada navio i, i = 1,...,n) FAÇA
3.      SE (berço que atende o navio i em S' ≠ berço que atende o navio i em c_j);
4.      REMOVER (o navio i de seu respectivo berço em S');
5.      INSERIR (o navio i no berço em S' correspondente ao berço em c_j);
6.      SE (f(S') < f(c_j)) ENTÃO
7.          c_j ← S';
8.      SENÃO
9.          SE (f(S') = f(c_j)) ENTÃO
10.             PARE;
11.         FIM-SE;
12.     FIM-SE;
13. FIM-PARA;
```

Figura 7. *Path-Relinking* utilizado na atualização dos centros de *clusters*.

A perturbação apresentada na linha 8 do EXECUTAR-CS (Figura 6) é dada por uma simples aplicação do movimento trocar navios. Por fim, a busca local (linha 11 do EXECUTAR-CS – Figura 6) utilizada para intensificar a busca em clusters promissores é apresentada na Figura 8. É interessante destacar que a busca local é aplicada por berço, evitando assim um alto tempo de processamento.

```
1.  PARA (cada berço pertencente a c_j) FAÇA
2.      ENQUANTO (melhorar a solução) FAÇA
3.          PARA (todos os navios de c_j) FAÇA
4.              INSERIR (o navio p em todas as posições do berço);
5.              ARMAZENAR (os navios e as posições que resultam na melhor solução);
6.          FIM-PARA;
7.          INSERIR (os navios nas melhores posições armazenadas);
8.      FIM-ENQUANTO;
9.  FIM-PARA;
```

Figura 8. Busca local.

6. Experimentos computacionais

Foram utilizadas 30 instâncias distintas, cada uma com 60 navios e 13 berços. Essas instâncias são baseadas em dados do porto de Gioia Tauro (Itália), e foram geradas aleatoriamente por Cordeau *et al*. (2005). Todos os experimentos foram realizados em um PC com processador AMD Athlon™ 64 de 2.2 GHz e 1GB de memória RAM (mesma máquina utilizada por Mauri *et al*. 2008a,b). Toda a implementação foi desenvolvida na linguagem C++.

Os parâmetros utilizados pelo CS, em todos os experimentos, foram T_0 = 20000, α = 0.975, T_C = 0.01, SAmax = 1000, λ= 7, rmax = 3 e o número de clusters foi igual 10. As penalizações utilizadas em ambos os casos foram ω = [1,10,10].

Foram realizados 5 testes para cada instância. A Tabela 1 apresenta os resultados obtidos nesses testes. A coluna Melhor f(S) apresenta a melhor solução (FO) encontrada nos cinco testes para cada instância. A coluna f(S) média apresenta a média aritmética das 5 funções objetivo encontradas, e a coluna Tempo Médio apresenta o tempo médio para resolver cada instância (em segundos). Por fim, a coluna Desvio é obtida pela equação abaixo.

$$\text{Desvio} = \left(\frac{f(S)\text{média} - \text{Melhor } f(S)}{\text{Melhor } f(S)} \right) * 100$$

Analisando a Tabela 1, percebe-se a robustez do CS, pois o método foi capaz de obter as soluções nos 5 testes em tempos computacionais baixos (média de 12,79 seg. por instância) com um desvio médio de 0,04%.

Tabela 1. Testes realizados com o CS.

Inst.	TESTE 1		TESTE 2		TESTE 3		TESTE 4		TESTE 5		Melhor	f(S)	Desvio	Tempo
	FO	Tempo	FO	Tempo	FO	Tempo	FO	Tempo	FO	Tempo	f(S)	Média	(%)	Médio
i01	1412	12,56	1409	12,44	1412	12,42	1413	12,45	1411	12,48	1409	1411,40	0,17	12,47
i02	1261	12,59	1261	12,55	1261	12,59	1261	12,62	1261	12,62	1261	1261,00	0,00	12,59
i03	1129	12,59	1129	12,66	1129	12,64	1129	12,66	1129	12,67	1129	1129,00	0,00	12,64
i04	1302	12,64	1302	12,59	1302	12,56	1302	12,59	1302	12,59	1302	1302,00	0,00	12,59
i05	1207	12,73	1207	12,69	1207	12,66	1207	12,66	1207	12,67	1207	1207,00	0,00	12,68
i06	1261	12,53	1261	12,59	1261	12,56	1261	12,53	1261	12,61	1261	1261,00	0,00	12,56
i07	1279	12,67	1279	12,61	1279	12,62	1279	12,61	1279	12,62	1279	1279,00	0,00	12,63
i08	1299	12,56	1299	12,58	1299	12,55	1299	12,61	1299	12,56	1299	1299,00	0,00	12,57
i09	1444	12,56	1444	12,72	1444	12,58	1444	12,52	1444	12,52	1444	1444,00	0,00	12,58
i10	1213	12,61	1213	12,62	1213	12,62	1213	12,56	1213	12,62	1213	1213,00	0,00	12,61
i11	1368	12,61	1368	12,56	1368	12,55	1369	12,59	1369	12,61	1368	1368,40	0,03	12,58
i12	1325	12,55	1325	12,53	1325	12,50	1325	12,61	1325	12,62	1325	1325,00	0,00	12,56
i13	1360	12,59	1360	12,66	1360	12,56	1360	12,67	1360	12,55	1360	1360,00	0,00	12,61
i14	1233	12,64	1233	12,64	1233	12,70	1233	12,72	1233	12,67	1233	1233,00	0,00	12,67
i15	1295	14,77	1295	15,89	1295	12,80	1295	12,77	1295	12,78	1295	1295,00	0,00	13,80
i16	1365	12,69	1365	12,69	1364	13,95	1364	16,42	1365	16,55	1364	1364,60	0,04	14,46
i17	1283	16,80	1283	13,45	1283	12,81	1283	12,77	1283	12,81	1283	1283,00	0,00	13,73
i18	1345	12,62	1345	12,59	1345	12,69	1345	13,06	1345	12,62	1345	1345,00	0,00	12,72
i19	1371	12,62	1368	13,23	1368	15,78	1367	12,86	1368	12,47	1367	1368,40	0,10	13,39
i20	1328	12,81	1329	12,83	1329	12,70	1328	12,81	1328	12,95	1328	1328,40	0,03	12,82
i21	1343	12,66	1343	12,72	1342	12,70	1341	12,70	1343	12,62	1341	1342,40	0,10	12,68
i22	1332	12,66	1326	12,58	1326	12,59	1328	12,67	1328	12,62	1326	1328,00	0,15	12,62
i23	1266	12,61	1266	12,64	1266	12,59	1266	12,64	1266	12,61	1266	1266,00	0,00	12,62
i24	1260	12,64	1260	12,62	1260	12,66	1260	12,66	1260	12,64	1260	1260,00	0,00	12,64
i25	1382	12,61	1376	12,69	1377	12,62	1377	12,62	1378	12,56	1376	1378,00	0,15	12,62
i26	1318	12,61	1323	12,64	1318	12,58	1324	12,62	1318	12,64	1318	1320,20	0,17	12,62
i27	1261	12,66	1261	12,66	1261	12,59	1261	12,69	1261	12,61	1261	1261,00	0,00	12,64
i28	1359	13,19	1360	12,58	1360	12,62	1360	12,62	1360	12,56	1359	1359,80	0,06	12,71
i29	1280	12,52	1281	12,55	1280	12,72	1281	12,64	1281	12,66	1280	1280,60	0,05	12,62
i30	1344	12,59	1345	12,61	1345	12,56	1344	12,56	1349	12,58	1344	1345,40	0,10	12,58

7. Considerações finais

Este trabalho teve por finalidade estudar o Problema de Alocação de Berços (PAB). Dentro desse contexto, buscou-se contribuir no aprimoramento de uma logística na distribuição do espaço do cais minimizando o tempo total de serviço dos navios, evitando assim prejuízos para o porto com embarcações rejeitadas.

Para resolver o PAB, foi proposta uma aplicação do método híbrido *Clustering Search* (CS) utilizando o *Simulated Annealing* como gerador de soluções. O CS mostrou ser adequado e eficiente na localização de regiões promissoras por meio do enquadramento dessas em clusters. Dessa forma, percebe-se que o CS atuou como uma alternativa para acelerar a obtenção de boas soluções.

De uma forma geral, os resultados obtidos demonstram que o CS foi capaz de gerar soluções de boa qualidade para todas as instâncias em tempos computacionais expressivamente baixos.

Agradecimentos: Os autores agradecem à FAPES (processo 45391998/09), ao CNPq (processo 471837/2008-3) e a CAPES pelo apoio financeiro.

Referências

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

Brown, G. G., Lawphongpanich, S. e Thurman, K. P. (1994), Optimizing ship berthing, *Naval Research Logistics*, 41, 1-15.

Brown, G. G., Cormican, K. J., Lawphongpanich, S. e Widdis, D. B. (1997), Optimizing submarine berthing with a persistence incentive, *Naval Research Logistics*, 44, 301-318.

Buhrkal, K., Zuglian, S., Ropke, S., Larsen, J. e Lusby, R., Models for the discrete berth allocation problem: a computational comparison. *Technical Report 14/2009 - Technical University of Denmark*, 2009.

Chaves, A. A., Metaheurísticas híbridas com busca por agrupamentos para problemas de otimização combinatória. *Tese* (Doutorado em Computação Aplicada) - Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2009.

Cheong, C. Y., Tan, K. C., Liu, D. K. e Lin, C. J. (2008), Multi-objective and prioritized berth allocation in container ports. *Annals of Operations Research*, [in press].

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 (8), 928-936.

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

Giallombardo, G., Moccia, L., Salani, M. e Vacca, I. (2010), Modeling and solving the tactical berth allocation problem. *Transportation Research Part B*, 44 (2), 232-245.

Hamming, R. W. (1950), Error detecting and error correcting codes. *Bell System Technical Journal*, 26(2),147-160.

Ilog. *ILOG CPLEX 10.0: user's manual*. France: [s.n.], 478 p, 2006.

Imai, A., Nishimura, E. e Papadimitriou, S. (2001), The dynamic berth allocation problem for a container port, *Transportation Research Part B*, 35, 401- 417.

Imai, A., Nishimura, E. e Papadimitriou, S. (2003), Berth allocation with service priority, *Transportation Research Part B*, 37, 437-457.

Imai, A., Sun , X., Nishimura, E. e Papadimitriou, S. (2005), Berth allocation in a container port: using a continuous location space approach. *Transportation Research Part B*, v. 39, n. 3, p. 199-221.

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

Mauri, G. R., Oliveira, A. C. M. e Lorena, L. A. N. (2008b), A hybrid column generation approach for the berth allocation problem, *Lecture Notes in Computer Science*, 4972, 110-122.

Oliveira, A. C. M. Algoritmos evolutivos híbridos com detecção de regiões promissoras em espaços de busca contínuo e discreto. *Tese* (Doutorado em Computação Aplicada) - Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2004.

Thurman, K. P., Optimal ship berthing plans. *Dissertação* (Masters of Science in Operations Research) - Naval Postgraduate School, Monterey, California - EUA, 1989.

UNCTAD. United nations conference on trade and development. *Review of maritime transport*, 2009.