GRASP Aplicado Ao Problema de Alocação de Berços em Terminais Portuários Graneleiros

Alexandre C. Muniz de Oliveria[1], **Igor L. C. Lima**[1],
Marcelo B. do Nascimento[1],
Dayson Nywton C. R. do Nascimento[1]

[1]Departamento de Ciência da Computação – Universidade Federal do Maranhão (UFMA)
São Luis, Brasil

acmo@deinf.ufma.br

Abstract. *In port terminals, ship's awaiting time is one of the main operational problems that may apply. Assigning a ship to an available berth is a decision which considers specific elements such as loading/unloading time, stock in the port's yard, fines for delays, and tide changes. In order to better use the resources offered by the port, the sequence of berths assigned to each ship must be planned. That planning is based on a queue of ships with their arrival time. To solve it, it is defined a sequence of ships to be allocated to a berth and the result is: less time of total operation, considering loading/unloading and awaiting time. In this paper presents the meta-heuristic GRASP (Greedy Randomized Adaptive Search Procedure) as a minimizing solution to the problem. The results show that the GRASP is a robust algorithm to solve this problem.*

Resumo. *Em terminais portuários, o tempo de espera dos navios é um dos principais problemas operacionais identificados. Atracar o navio em um berço disponível é uma decisão que leva em conta fatores como: seu tempo de carga/descarga, estoque no pátio, multas sobre atrasos, além de influência de marés. O planejamento, para melhor aproveitamento de recursos do porto, baseia-se na ideia da existência de uma fila de navios, onde são conhecidos os seus tempos aproximados de chegada. Resolver o problema é definir uma sequência de atracação que resulte em menor tempo de operação total do porto, levando em consideração os tempos de carga/descarga e espera. Este artigo propõe a utilização da meta-heurística GRASP (Greedy Randomized Adaptive Search Procedure) como solução de minimização do tempo total de serviço que o navio passa no atendimento do berço. Os resultados mostram que o GRASP é um algoritmo robusto para resolver esse problema.*

1. Introdução

No Maranhão, o complexo portuário marítimo de São Luís, formado pelos terminais do Itaqui, Ponta da Madeira e da ALUMAR (Consórcio de Alumínio do Maranhão) são responsáveis pela segunda maior movimentação de carga no Brasil. A ALUMAR movimenta matérias primas como coque, piche, soda e bauxita, empregados no processo de produção de alumina e alumínio. A mineradora Vale, por sua vez, detém a administração do terminal da Ponta da Madeira por onde é exportado minério de ferro proveniente da jazida da Serra de Carajás. Além desses, o Porto do Itaqui, administrado pela Empresa

Maranhense de Administração Portuária (EMAP), movimenta vários tipos de cargas, tais como: derivados de petróleo, fertilizante, manganês, ferro gusa, e soja [Barros 2010].

Os portos tem importante valor na competição econômica entre os países. É pelas águas marinhas que se escoa mais da metade da produção mundial. Os navios tendem a aumentar de capacidade com o avanço da tecnologia, exigindo que os portos se adequem cada vez mais a esses avanços. É nesse cenário que surge a necessidade de explorar a maior capacidade possível do porto e a diminuição dos custos de espera e manutenção [Silva 2008]. O porto ideal seria capaz de atender navios de grande porte, com alto nível de mecanização, oferecendo fluidez ao transporte e elevada produtividade [Barros 2010].

Os complexos portuários de todo o mundo tem por objetivo fundamentalmente realizar carga e descarga de navios em locais para tais propósitos conhecidos como berços [Barros 2010]. Para essas operações acontecerem com eficiência, existe uma série de equipamentos que precisam estar disponíveis e condições que precisam ser cumpridas.

O Problema de Alocação de Berços (PAB) consiste em minimizar o tempo total das operações no porto. Dada uma fila de navios e conhecidos os seus tempos estimados de chegada, o problema se resume a definir uma sequência de atracação que resulte no menor tempo de serviço de cada navio. Este trabalho tem por objetivo propor a meta-heurística GRASP para a resolução do PAB, com condições restritas ao complexo portuário do Itaqui.

O restante deste trabalho está organizado da seguinte forma: o Problema de Alocação de Berços é descrito na Seção 2; na Seção 3 é apresentado o GRASP e como ele foi aplicado ao PAB; os resultados são discutidos na Seção 4; e na Seção 5 são tiradas algumas conclusões em relação a utilização do GRASP para este problema.

2. Problema de Alocação de Berços

O problema de alocação de berços com restrição de estoque foi modelado levando em consideração a quantidade de navios N e a quantidade de marés L, com base em uma análise discreta do tempo em relação a janelas de horizonte M chamadas de *Tidal Time Windows (TTW)*, como mostra a Figura 1.

A fundamentação matemática foi baseada no modelo de heterogeneidade dos berços [Barros 2010], onde cada berço tem uma velocidade de atendimento diferente nas operações com os navios. Este fator de heterogeneidade está diretamente ligado à infraestrutura dos berços, pois diferentes estruturas resultam em vazões ou velocidades de atendimentos diferentes.

Nesse modelo, "o objetivo é minimizar o custo acumulado sobre todas as operações em um horizonte de planejamento dado. Sendo assim, apenas o tempo de serviço total é utilizado, obtido com base na utilização de TTW's por cada navio"[Barros 2010].

A função objetivo consiste em minimizar os tempos de serviço de cada navio, como mostra a fórmula a seguir:

$$min \sum_{i=1}^{N} \sum_{j=a_i}^{M} \sum_{l=1}^{L} \left\lceil \frac{j - a_i + 1}{h_{ij}} \right\rceil \times y_{ijl} \tag{1}$$

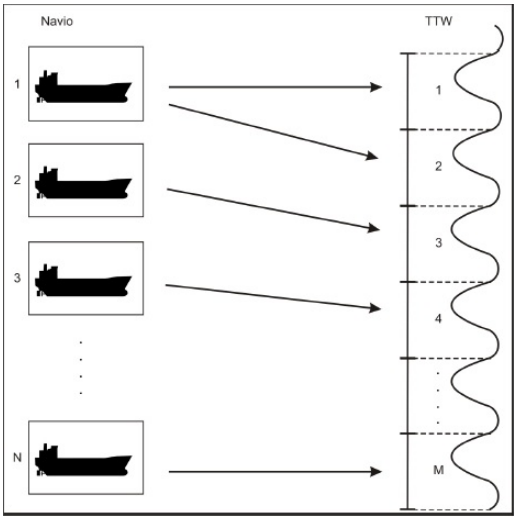

Figura 1. Demonstração do PAB. [Barros 2010]

Onde:

- N: número de navios, $n = |N|$;
- M: número de TTW's, $m = |M|$;
- L: conjunto de posições de atracação, $l = |L|$
- a_i: TTW de chegada do navio i;

A variável de decisão assume os seguintes valores:

$$y_{ijl} = \begin{cases} 1, \ se\ o\ navio \text{ i } estiver\ alocado\ na\ TTW \text{ j } e\ atracado\ no\ berço \text{ l} \\ 0, \ de\ outra\ forma \end{cases}$$

Na modelagem do problema é utilizado um parâmetro que consiste no tempo de processamento de um navio em um berço, representado pela variável h_{il}, que é descrito da seguinte forma:

$$h_{il} = \left\lceil \frac{\sum_{q=1}^{|Q|} w_{iq}}{v_l} \right\rceil \tag{2}$$

No qual v_l representa o tempo de carregamento/descarregamento [Fernandes 2001] no berço l, Q representa a quantidade total de materias-primas operadas no porto e w_{iq} a quantidade de matéria-prima q em um navio i.

3. GRASP

O GRASP é aplicado em problemas de otimização combinatória onde cada iteração consiste em duas fases: a construção de uma solução inicial de forma gulosa, aleatória e adaptativa e um procedimento de busca local que tem o objetivo de aprimorar a solução gerada na primeira fase [Gonçalves 2004]. Apesar de semelhante a várias outras heurísticas gulosas, o GRASP diferencia-se pela geração da primeira solução aleatória, que é realizada por algum tipo de construção randômica gulosa ou algum esquema de perturbação [Freddo 2008].

O GRASP, apesar de ser uma heurística gulosa, ainda tem sido muito utilizado para resolver grandes problemas de otimização combinatória, sendo, segundo [Lopes 2013], responsável por produzir algumas das melhores soluções conhecidas para muitos problemas. Outro ponto importante é a fácil implementação da meta-heurística, que necessita de apenas dois parâmetros iniciais ι e α, sendo ι responsável pelo número de iterações e α pela taxa de "gulosidade" do esquema construtivo. Segue um esquema do pseudo algoritmo GRASP.

Para aplicar o GRASP ao PAB, é necessário que se definam os esquemas de busca local e a heurística gulosa a ser utilizada. A seguir, serão abordados os tópicos referente a aplicação do algoritmo exposto no trabalho ao problema.

Algoritmo 1: GRASP

```
1  enquanto houver navios a serem alocados faça
2        x <- geraSoluçãoAleatóriaGulosa();
3        buscaLocal(x);
4  se x < x* então
5        x* <- x;
```

3.1. Algoritmo Construtivo Guloso Aleatório

O GRASP se utiliza de uma heurística aleatória e gulosa para a construção da solução inicial. Existem várias formas de se chegar a uma boa heurística para um dado problema, uma delas é a Construção Semigulosa proposta por [Lopes 2013]. Essa construção foi escolhida pela fácil adaptação ao problema aqui tratado, porém outras construções são sugeridas pelo mesmo autor. A Construção Semigulosa é uma heurística baseada em uma lista restrita de candidatos chamada de LRC. O algoritmo funciona da seguinte forma: a cada iteração um novo elemento, dentre os mais promissores, é adicionado à solução, alterando dessa forma o cenário; novamente são estimados os pesos dos candidatos e um, dentre os mais promissores, é adicionado à solução. A construção é adaptativa por se tratar de um algoritmo que calcula novos pesos para cada nova inserção de candidato, redefinindo assim os valores e pesos, e adaptando-se ao problema. É aleatória porque o próximo candidato a entrar na lista é escolhido aleatoriamente dentre os mais promissores.

Uma função avaliadora estima o peso de todos os candidatos da LRC baseado no impacto que irá causar ao entrar na solução naquele ponto da construção. Após avaliar cada candidato, é feita uma seleção aleatória, onde o parâmetro α é o ponto de corte dessa seleção. Estarão aptos a entrar na solução somente os α primeiros candidatos da lista. Esse parâmetro, como já dito anteriormente, determina a "gulosidade" do algoritmo. O termo semiguloso é aplicado devido ao fato de o algoritmo utilizar-se de uma função estimadora, um percentual de corte e uma lista restrita. Portanto, por mais que seja guloso, o algoritmo utiliza-se de uma estratégia para fugir de soluções muito ruins. Um esquema do pseudo algoritmo da construção gulosa é mostrado a seguir.

Algoritmo 2: Algoritmo Construtivo Guloso Aleatório

```
1  enquanto lista de candidatos não for vazia faça
2        avaliaCandidatos(C);
3        ci = selecionaAleatoriamenteUmCandidato(C);
4        adicionaCandidatoNaSolução(ci);
```

Para aplicar a Construção Semigulosa ao PAB é necessário que se defina a função avaliadora que será capaz de estimar pesos para os candidatos da LRC. Sendo assim, a função avaliadora para este trabalho baseia-se na quantidade de carga trazida por cada navio relacionada às marés críticas de suas matérias.

Entende-se como um bom navio a entrar na solução aquele que traz maior quantidade de matéria e o consumo dessa matéria pelo porto é alto. Além disso, um esquema de matérias críticas é tratado no algoritmo. A criticidade de uma matéria prima é calculada em função de seu estoque atual e o consumo. A cada TTW um consumo pelo porto é realizado sobre as matérias no estoque. Em uma determinada TTW o consumo da matéria irá superar o seu nível de estoque, resultando no que aqui é chamada de maré crítica. Todas as matérias terão uma maré crítica, que é aquela TTW em que a quantidade da matéria estará em um nível abaixo do requerido pelo porto. Para cada navio é estimada, no caso de trazer mais de uma matéria, a sua maré mais crítica. Portanto, o potencial de entrada do navio na construção semigulosa é avaliado como um fator proporcional à criticidade da matéria da carga trazida por ele. O potencial do navio aumenta com a proximidade da maré (tempo) na qual o estoque atinge o valor mínimo (maré crítica). A seguir, está definida a função avaliadora, onde d é a distância da maré de chegada do navio à maré crítica da matéria prima que ele carrega, e e é a quantidade de carga trazida por ele. Vale ressaltar que $g(c_k)$ avalia a matéria k de cada navio resultando em pesos diferentes quando o navio estiver carregado com mais de uma matéria. Nesse caso, aquela matéria que apresentar maior peso determinará o peso total do navio.

$$g(c_k) = \frac{1}{d} \times e \tag{3}$$

3.2. Busca Local

Um esquema de busca local pode ser definido como uma estrutura de vizinhança, onde cada vizinho é obtido por um movimento na solução [Freddo 2008]. Esses movimentos podem ser pequenas alterações na solução, como relata [Lopes 2013]. A busca local parte de um ponto no espaço de solução à procura de soluções vizinhas melhores que a atual onde, encontrada uma solução melhor, a busca reinicia recursivamente sobre a mesma.

Algoritmo 3: Busca Local

```
1  enquanto existir s pertencente a N(t) tal que f(s) < f(t) faça
2        t <- s;
```

No caso específico do PAB, um esquema de busca local é definido como movimentos de trocas ou alterações no vetor solução. Dada uma solução, buscar um vizinho desta significa alterar posições de navios ou trocar berços entre dois ou mais navios. Esses movimentos produzem outras soluções que podem ser melhores ou piores que a solução corrente. Outros esquemas de buscas locais mais elaborados podem ser tratados no PAB, mas este não é o enfoque deste trabalho.

4. Resultados

Para concluir e confirmar as pesquisas deste trabalho, foi necessária a realização de diversos testes com instâncias do PAB. Os resultados consolidaram a heurística gulosa, que se mostrou efetiva e apresentou boas soluções, conseguindo alcançar a solução ótima em muitos casos. O esquema de construção gulosa gerou boas soluções iniciais. Além disso, o procedimento de busca local mostrou-se robusto e bastante eficiente na exploração do espaço de busca.

Tabela 1. Resultados computacionais obtidos por Greedy Randomized Adaptive Search Procedure (GRASP).

Instância	Navios	Berços	TTW's	Ótimo	Média	Desvio Padrão	Ótimo (GRASP)	Taxa
10-15-3A	10	3	15	31	31	0	31	1
10-15-4A	10	4	15	29	30	0	30	0
10-15-4B	10	4	15	21	21	0	21	1
10-20-3A	10	3	20	44	44	0	44	1
10-20-3B	10	3	20	27	27	0	27	1
10-25-2A	10	2	25	43	44	0	44	1
10-25-2B	10	2	25	33	83	0	83	1
15-20-4A	15	4	20	40	40	0	40	1
15-25-3A	15	3	25	55	55	0	55	1
15-30-2A	15	2	30	-	110	0	110	-
15-30-4A	15	4	30	77	77,93	0,26	77	0,066
15-35-3A	15	3	35	86	87,2	0,81	86	0,233
15-40-2A	15	2	40	121	125	0	125	0
20-40-3A	20	3	40	107	110,46	0,74	109	0
20-40-4A	20	4	40	108	108,26	0,46	108	0,733
20-55-2A	20	2	55	297	319,7	1,52	317	0
30-70-4A	30	4	70	-	208,86	1,97	206	-
30-75-3A	30	3	75	208	216,76	1,87	212	0
30-80-2A	30	2	80	-	809,26	30,81	772	-

As instâncias do problema têm tamanhos variados para avaliar de maneira adequada a performance do algoritmo, sendo essas compostas por dados de entrada baseados em estimativas de condições reais do porto. Para a produção dessas instâncias foram tomados horizontes de planejamentos previamente estudados e estruturados, que possibilitassem a resolução computacional com o ideal de aproximar os dados tratados pelo algoritmo do cenário real. Os resultados ótimos conhecidos foram obtidos por programas solucionadores comerciais, porém algumas instâncias possuem valores ótimos desconhecidos, como no caso das instâncias "15-30-2A","30-70-4A" e"30-80-2". A tabela de resultados (Tabela 1) abaixo foi obtida após a execução do algoritmo, fazendo uso das instâncias citadas acima. Além disso, apresenta dados como média - resultado médio alcançado - taxa - porcentagem de resultados econtrados igual ao ótimo - e ótimo (GRASP) - melhor (e consequentemente menor) valor encontrado por esse algoritmo para aquela instância. Essas informações são necessárias uma vez que o GRASP é um algoritmo estocástico,

gerando, dessa forma, soluções variadas em cada iteração e que serve, portanto, para que se verifique o quão eficaz ele se mostrou.

Os tempos de execução foram abaixo do estipulado para a heurística, que é de 3600 segundos, e cada instância foi executada trinta vezes.

5. Conclusão

Este artigo apresentou uma metodologia de resolução de problemas de otimização combinatória tomando o PAB como objeto de estudo e aplicação do método (GRASP). Foram tratados os cenários do problema, bem como o algoritmo e esquema utilizados na resolução do mesmo. Os resultados do algoritmo mostraram-se bons reflexos da heurística gulosa proposta pelo problema, bem como do esquema de busca local e atribuição de pesos aos navios.

O presente trabalho pode ainda estender-se visto que grandes instâncias do PAB ainda podem ser resolvidas pelo GRASP. Algumas mudanças na busca local e na heurística gulosa podem trazer resultados ainda melhores, como a dissolução da função objetivo no algoritmo, que seria dar o fitness da solução em tempo de construção da mesma. Até o presente trabalho as pesquisas não avançaram devido a complexidade dos estudos. Pode-se ainda testar as variações de GRASP que tiveram bons resultados em outros trabalhos, como cita [Lopes 2013].

Outro ponto importante a ser ressaltado é a melhoria na geração da solução inicial. Foi constatado que se houver um método para unir a heurística gulosa à função objetivo, podem-se obter soluções que se utilizam de táticas para um melhor ordenamento dos navios, bem como atrasar um navio para gerar um ganho na solução total.

O GRASP conseguiu mostrar a eficiência da metaheurísitica para resolver problemas de otimização linear, uma vez que produziu soluções muito boas e com baixo custo computacional.

Referências

Barros, V. H. (2010). Problema de alocação de bercos em portos graneleiros com restricão de estoque e condições favoraveis de maré. Dissertação (Mestrado em Engenharia Elétrica), UFMA, São Luis, MA.

Fernandes, M. G. (2001). Modelo econômico-operacional para análise e dimensionamento de terminais de contêineres e veículos, USP, São Paulo.

Freddo, A. R., . B. R. C. (2008). Implementação da metaheurística grasp para o problema do caixeiro viajante simétrico, Universidade Federal do Paraná.

Gonçalves, L. B., M. S. L. . O. L. (2004). Uma heurística grasp para o problema do caixeiro viajante periódico. *Anais do XXXVI SBPO*.

Lopes, H. S., R. L. C. . S. M. T. (2013). *Meta-heurísticas em pesquisa operacional.* Omnipax, Curitiba, PR.

Silva, V. M. (2008). Um modelo heurístico para alocação de navios em berços. Dissertação (Mestrado em Engenharia Elétrica), USP, São Paulo.