UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO DE CIÊNCIAS EXATAS, NATURAIS E DA SAÚDE
DEPARTAMENTO DE COMPUTAÇÃO

ALCEBIADES LOPES GAMAS VENTURELI

META-HEURÍSTICA HÍBRIDA GRASP E *SIMULATED ANNEALING* PARA O PROBLEMA DE ALOCAÇÃO DE BERÇOS

ALEGRE - ES
2023

ALCEBIADES LOPES GAMAS VENTURELI

META-HEURÍSTICA HÍBRIDA GRASP E *SIMULATED ANNEALING* PARA O PROBLEMA DE ALOCAÇÃO DE BERÇOS

Trabalho de conclusão de curso apresentado ao Departamento de Computação do Centro de Ciências Exatas, Naturais e da Saúde da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Edmar Hell Kampke.

ALEGRE - ES

2023

ALCEBIADES LOPES GAMAS VENTURELI

META-HEURÍSTICA HÍBRIDA GRASP E SIMULATED ANNEALING PARA O PROBLEMA DE ALOCAÇÃO DE BERÇOS

Trabalho de Conclusão de Curso apresentado ao Departamento de Computação do Centro de Ciências Exatas, Naturais e da Saúde da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Aprovado em 21 de julho de 2023.

COMISSÃO EXAMINADORA

Prof. Dr. Edmar Hell Kampke
Universidade Federal do Espírito Santo
Orientador

Prof. Dr. Geraldo Regis Mauri
Universidade Federal do Espírito Santo

Prof. Dr. Dayan de Castro Bissoli
Universidade Federal do Espírito Santo

AGRADECIMENTOS

Agradeço primeiramente ao Todo Poderoso Deus de Israel, pelo cuidado, graça e misericórdia. Sem o Eterno, não teria chegado até aqui, seja louvado o Seu Nome para sempre!

À minha mãe Marlene Lopes Gamas, ao meu pai Alcebides da Silva Gamas e aos meus irmãos, pelo amor, carinho e esforço em me manter estudando todos esses anos, bem como, pelo apoio, incentivo e orações.

À minha esposa Núbia Ventureli Machado Gamas que sempre esteve ao meu lado durante toda a minha jornada, sendo motivadora, inspiradora e amorosa.

Ao meu amigo, conselheiro, padrinho de casamento e orientador Prof. Dr. Edmar Hell Kampke, pela bondade em ajudar-me, pela paciência e por todas as contribuições e direcionamentos.

Aos meus amigos, conselheiros e padrinhos de casamento Prof. Dr. Christiano Jorge Gomes Pinheiro e sua esposa Natália Ferraz Thomé de Azevedo, pelo carinho, conselhos, orações e incentivos.

À minha sogra Marilda Ventureli Machado e ao meu sogro Ubirajara Machado de Oliveira pelo apoio, incentivo e orações.

Ao meu pastor Claudemir Cruz de Almeida e sua esposa Lucinéia Rezende Gomes de Almeida, pelas orações e palavras de sabedoria.

À Universidade Federal do Espírito Santo e todos os seus funcionários, principalmente os professores do Departamento de Computação, que puderam transmitir seus conhecimentos.

Aos meus amigos que de alguma forma contribuíram com a minha formação.

RESUMO

Grande parte do envio de mercadorias do comércio internacional é realizado através do transporte marítimo. Os portos, por sua vez, precisam realizar o planejamento da atracação dos navios para atendê-los de forma rápida e eficiente, a fim de evitar atrasos nas entregas ou mesmo, multas de contrato com as empresas que fazem uso dos seus serviços, bem como satisfazer quaisquer restrições de atendimento. Na literatura, este problema é conhecido como Problema de Alocação de Berços (PAB) e neste trabalho, é proposta uma abordagem baseada na aplicação do método *Greedy Randomized Adaptive Search Procedure* (GRASP), de forma integrada com o método *Simulated Annealing* (SA), para resolução do PAB. A abordagem proposta consiste na aplicação do GRASP, que a cada iteração constrói uma solução e a submete ao SA, usado neste trabalho como estratégia de busca local. Por meio dessa abordagem, foram obtidas boas soluções e de forma rápida para cinco instâncias do PAB que foram obtidas na literatura.

Palavras-chave: Portos, Problema de Alocação de Berços, *Simulated Annealing*, GRASP.

ABSTRACT

Much of the shipment of goods in international trade is carried out by sea transport. The ports, in turn, need to plan the berthing of ships to serve them quickly and efficiently, to avoid delays in deliveries or even contract fines with companies that make use of their services, as well as are subject to service restrictions. In the literature, this problem is known as the Berth Allocation Problem (BAP) and in this work, an approach is proposed based on the application of the Greedy Randomized Adaptive Search Procedure (GRASP) method integrated with the Simulated Annealing (SA) method for BAP resolution. The proposed approach consists of applying GRASP, which at each iteration creates a solution and submits it to the SA, used in this work as a local search strategy. Through this approach, good solutions were initiated quickly for five instances of the BAP that were initiated in the literature.

Keywords: Ports, Berth Allocation Problem, Simulated Annealing, GRASP.

LISTA DE FIGURAS

Figura 1 - Cenário típico para o PAB. 12
Figura 2 - Representação PABD, PABC e PABH. 15
Figura 3 – Exemplo de sequência de atendimento de navios no PABD 15
Figura 4 - Pseudocódigo genérico da meta-heurística Simulated Annealing 17
Figura 5 - Pseudocódigo genérico da meta-heurística GRASP 19
Figura 6 - Variáveis referentes ao tempo. 21
Figura 7 - Representação de uma solução. 23
Figura 8 - Pseudocódigo da heurística construtiva aleatória gulosa. 24
Figura 9 - Movimento reordenar navios 25
Figura 10 - Movimento realocar navios 26
Figura 11 - Movimento trocar navios 26
Figura 12 - Pseudocódigo da meta-heurística proposta GRASP+SA. 27

LISTA DE TABELAS

Tabela 1 – Resultados obtidos. 30
Tabela 2 – Comparação de resultados. 31

LISTA DE ABREVIATURAS E SIGLAS

AG – Algoritmo Genético

AM – Algoritmo Memético

BI – *Best Improvement*

CCENS – Centro de Ciências Exatas, Naturais e da Saúde

DARP – *Dial-a-Ride Problem*

FI – *First Improvement*

GRASP – *Greedy Randomized Adaptive Search Procedure*

PAB – Problema de Alocação de Berços

PABC – Problema de Alocação de Berços com *Layout* Contínuo

PABD – Problema de Alocação de Berços com *Layout* Discreto

PABH – Problema de Alocação de Berços com *Layout* Híbrido

PCCP – Problema da Cadeia de Caracteres mais Próxima

PR – *Path Relinking*

RAM – *Random Access Memory*

SA – *Simulated Annealing*

UFES – Universidade Federal do Espírito Santo

VNS – *Variable Neighborhood Search*

SUMÁRIO

1. INTRODUÇÃO 10

1.1. O PROBLEMA E SUA IMPORTÂNCIA 11

1.2. OBJETIVOS 12

1.2.1. Objetivo Geral 12

1.2.2. Objetivos Específicos 13

2. REVISÃO DE LITERATURA 14

2.1. PROBLEMA DE ALOCAÇÃO DE BERÇOS 14

2.2. SIMULATED ANNEALING – SA 17

2.3. GREEDY RANDOMIZED ADAPTIVE SEARCH PROCEDURE - GRASP 19

3. METODOLOGIA 21

3.1. FORMULAÇÃO MATEMÁTICA 21

3.2. REPRESENTAÇÃO DE UMA SOLUÇÃO 22

3.3. HEURÍSTICA CONSTRUTIVA ALEATÓRIA GULOSA 23

3.4. ESTRUTURA DE VIZINHANÇA 25

3.5. GRASP+SA PARA O PAB 26

4. RESULTADOS 28

4.1. INSTÂNCIAS PARA O PAB 28

4.2. PARÂMETROS PARA O GRASP 28

4.3. PARÂMETROS PARA O SA 29

4.4. AMBIENTE DE TESTES 30

4.5. RESULTADOS COMPUTACIONAIS 30

5. CONCLUSÕES 32

REFERÊNCIAS BIBLIOGRÁFICAS 33

1. INTRODUÇÃO

De acordo com Hansen et al. (2008) a importância dos portos cresce à medida que o transporte marítimo aumenta, estimulado pelos avanços na tecnologia para construção de navios maiores. Além disso, a frequência do transporte aumenta conforme cresce o volume de comércio internacional gerado pelo crescimento econômico.

Segundo Zhenhub (2022), devido ao conflito entre Rússia e Ucrânia, somado a Pandemia da COVID-19, cerca de 77% dos portos do mundo estão enfrentando grandes atrasos para retomarem suas atividades. Dezenas de navios permanecem ancorados na costa de portos dos EUA e da China, ociosos e aguardando por espaço para atracar. Zhenhub (2022) ainda descreve que as cadeias de suprimentos globais passaram por um momento difícil desde 2020. Sua dependência do transporte marítimo é tão grande que 90% das mercadorias do mundo são transportadas em contêineres e navios.

Assim, no que se refere à economia mundial, podemos afirmar que esta é fortemente dependente do transporte marítimo, por ser o principal meio utilizado para a realização de importação e exportação de produtos. Consequentemente os serviços ofertados pelos portos precisam ser rápidos e eficientes, conforme descrito por Hansen et al. (2008).

Uma solução para o Problema de Alocação de Berços (PAB) consiste estabelecer uma sequência de atendimento aos navios nos berços de um porto qualquer, minimizando o tempo total de serviço. De maneira geral, esse problema na literatura é abordado como um problema de otimização combinatória. Para problemas dessa natureza, comumente são utilizadas meta-heurísticas para obtenção de soluções. Neste trabalho será utilizada uma metaheurística hibrida para o PAB.

1.1. O PROBLEMA E SUA IMPORTÂNCIA

Os navios que chegam a um determinado porto são distribuídos ao longo de um cais em “posições” de atracação disponíveis, essas são denominadas berços.

Um berço pode possuir restrições para atracação de determinados navios, pois cada berço possui restrições relativas à profundidade da água, à distância máxima em relação ao local mais favorável ao longo do cais e ao tamanho dos navios (MAURI, 2008). Assim, navios são colocados em espera, aguardando a disponibilidade de um berço capaz de atendê-lo.

Para Imai et al. (2003), a alocação de berços tem um impacto primordial na eficiência das operações realizadas pelos portos no tocante ao atendimento de navios. Trata-se, portanto, de um problema conhecido na literatura como Problema de Alocação de Berços (PAB). Dessa forma, o PAB consiste em alocar os navios que chegam no porto para os berços de atracação disponíveis ao longo de um cais (MAURI, 2008). A Figura 1 apresenta um típico cenário do PAB, nela podemos observar que um cais ao longo de sua extensão, possui uma quantidade limitada de berços para que navios possam atracar, além disso, a figura ilustra as diversas etapas que um navio deve passar até o seu atendimento:

a) O navio realiza o deslocamento até o porto;
b) Aguarda disponibilidade de berço capaz de atendê-lo, caso estejam totalmente ocupados.
c) O navio atraca em um berço vago que seja capaz de atendê-lo.
d) Após o término do atendimento, o navio segue viagem liberando o berço que estava ocupando para a atracação de outro navio.

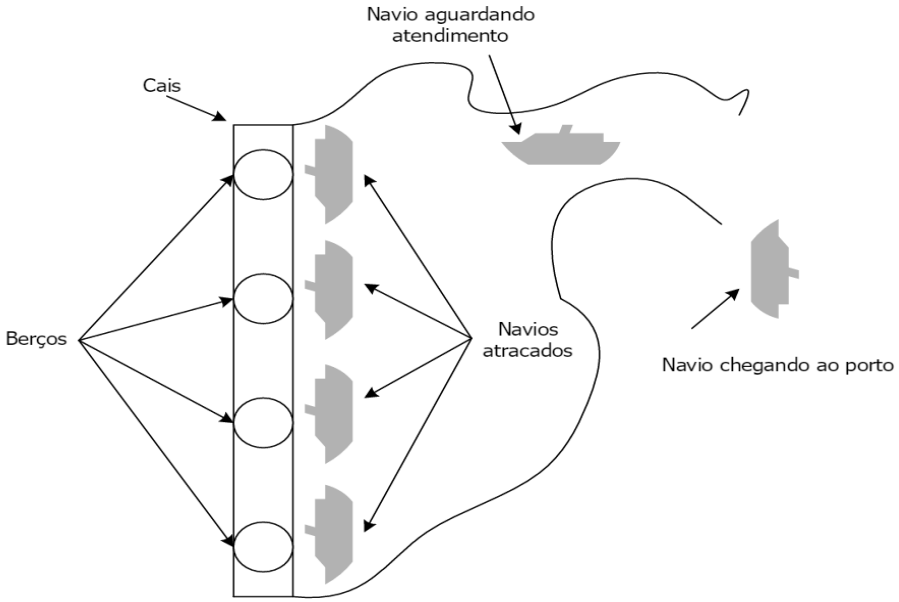

Figura 1 - Cenário típico para o PAB (MAURI, 2008).

Hansen et al. (2008) ainda expõe que os berços são os recursos mais importantes a serem utilizados para uma gestão econômica do porto, uma vez que uma boa alocação de berços para os navios aumenta a produtividade do porto, resultando em maiores receitas e satisfação, tanto para proprietários dos navios quanto para o porto. Além disso, o PAB reflete as principais decisões referente aos recursos de custo mais importantes no gerenciamento de portos de cargas (MAURI et al., 2010).

1.2. OBJETIVOS

Nesta seção são apresentados o objetivo geral, bem como os específicos deste trabalho.

1.2.1. Objetivo Geral

Este trabalho tem como finalidade a implementação da meta-heurística híbrida Greedy Randomized Adaptive Search Procedure (GRASP) com Simulated Annealing (SA) para encontrar soluções de qualidade e de forma eficiente para o PAB. A escolha dos algoritmos GRASP e SA para hibridização proposta neste trabalho se deu pela robustez de ambos, observando a ampla utilização deles em problemas de otimização combinatória presentes na literatura.

1.2.2. Objetivos Específicos

Além do objetivo geral, pretende-se atingir os seguintes objetivos específicos:

a) Implementar uma heurística construtiva com aleatoriedade e gulosidade.
b) Desenvolver a meta-heurística SA como estratégia de intensificação de busca.
c) Combinar esses métodos através da meta-heurística GRASP.
d) Executar testes iniciais para calibração de parâmetros.
e) Validar os resultados iniciais encontrados;
f) Corrigir e ajustar, se necessário, os métodos implementados.
g) Realizar testes finais com instâncias disponíveis na literatura.
h) Analisar os resultados obtidos comparando-os com outros trabalhos da literatura.

2. REVISÃO DE LITERATURA

Nesta seção são apresentados os fundamentos teóricos que embasam o desenvolvimento da solução proposta.

2.1. PROBLEMA DE ALOCAÇÃO DE BERÇOS

Segundo Mauri (2008), os principais trabalhos que abordam o PAB apresentam métodos heurísticos e isso se deve ao tratamento de casos reais existentes na maioria deles.

Na literatura, problemas desse tipo geralmente impõe dois tipos de restrições: espaciais e temporais, sendo que no primeiro tipo, elas estão relacionadas às dimensões do porto e dos navios, à profundidade do porto e à partição da área acostável, ou seja, fortemente ligadas ao local de atracação. Já no segundo, elas são principalmente relacionadas às datas de chegada e atracação dos navios (CORDEAU et al., 2005). Dessa forma, no que se refere às restrições espaciais no PAB quanto à definição do *layout* das áreas acostáveis do porto, podemos classificá-los em três tipos: PAB com *layout* discreto (PABD), PAB com *layout* contínuo (PABC) e PAB com *layout* híbrido (PABH) (RODRIGUES et al., 2016).

- PABD – Divide a área acostável em berços, onde apenas um navio pode atracar e ser atendido em um determinado intervalo de tempo, desde que seja respeitada a restrição de que o comprimento do navio não seja maior que o comprimento do berço.
- PABC – A área acostável não é dividida, dessa forma os navios podem atracar em qualquer seção ao longo do cais.
- PABH – A área acostável é também dividida em berços, no entanto existe a possibilidade que um navio ocupe mais de um berço, caso seja necessário, devido ao seu tamanho, bem como dois navios ocupem um único berço.

A Figura 2 ilustra esses tipos de PAB. Assim, pode-se ter uma maior compreensão dos tipos de *layouts*.

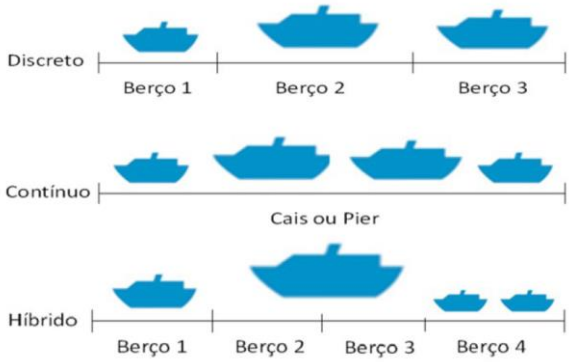

Figura 2 - Representação PABD, PABC e PABH (RODRIGUES et al., 2016).

Já em relação as restrições temporais, geralmente os processos de chegada do PAB são classificados como: chegada estática e chegada dinâmica, sendo que na abordagem estática considera-se que todos os navios se encontram prontos para atracar, ou seja, é desconsiderada a data de chegada dos navios. Na abordagem dinâmica os navios possuem datas de chegadas diferentes ao longo de um horizonte de planejamento e assim não é possível serem atendidos antes do seu horário de chegada ao porto (RODRIGUES et al., 2016). O PAB com chegada dinâmica retrata um cenário fiel ao que ocorre no dia a dia dos portos e as decisões que às respectivas administrações desses portos precisam tomar (BIERWIRTH e MEISEL, 2010; IMAI et al., 2001).

Ao unir os conceitos de restrições espaciais e temporais é esperado que a resolução do PAB determine a sequência de atendimento dos navios no berço. A Figura 3 exemplifica uma sequência de atendimento de oito navios, ao longo de uma área acostável que contém quatro berços, no contexto do PABD.

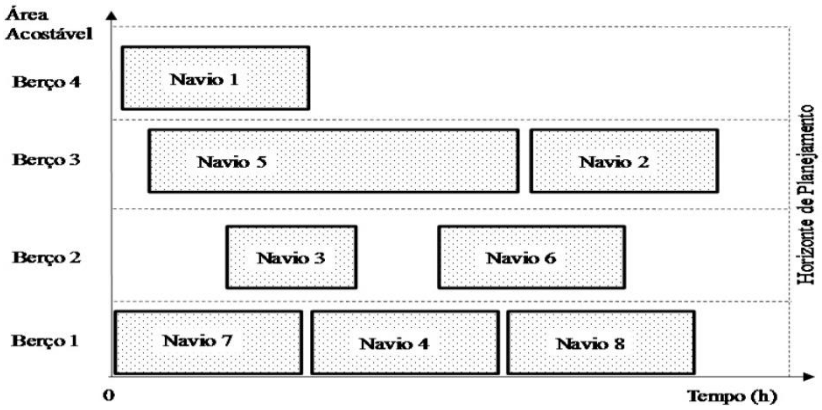

Figura 3 – Exemplo de sequência de atendimento de navios no PABD (RODRIGUES et al., 2016).

Thurman (1989) propôs o primeiro modelo de otimização para o PAB. Mais tarde ele é aprimorado por Brown et al. (1994) e os conflitos entre navios são minimizados. Brown et al. (1997) ainda propuseram um planejamento de berços para submarinos.

Cordeau et al. (2005) propuseram uma abordagem baseada na Busca Tabu para resolução do PAB para os casos discreto e contínuo. Os autores observaram que apenas instâncias pequenas podem ser resolvidas de forma exata e a Busca Tabu sempre encontra as mesmas soluções. Já para instâncias maiores os autores descrevem que os resultados obtidos são melhores quando comparados aos obtidos de uma versão *branch-and-bound*. Foram utilizados um grande conjunto de instâncias, a maioria gerada a partir de informações reais do porto de *Gioia Tauro* (Itália) e 30 delas obtidas de forma aleatória, com 60 navios e 13 berços (MAURI, 2008).

Hansen et al. (2008) propuseram uma heurística de busca em vizinhança variável (*Variable Neighborhood Search* - VNS) e a compararam com uma heurística VNS com inicializações múltiplas (*multi-start)*, um Algoritmo Genético (AG) e um Algoritmo Memético (AM), todas para o PABD. Foram utilizados conjuntos de instâncias entre cinco berços e 50 navios até 20 berços e 50 navios. O VNS forneceu boas soluções para todas as instâncias e elas foram comparadas com as obtidas em outro artigo dos autores. O desempenho dos algoritmos: VNS com *multi-start*, AG e AM, foi superado pelo algoritmo VNS principalmente em instâncias consideradas grandes.

Lopes et al. (2011) propôs uma abordagem baseada na aplicação do método GRASP integrada com o método *Path Relinking* (PR) para a resolução do PABD, utilizando o modelo proposto por Cordeau et al. (2005), combinado com relaxações de restrições propostas por Mauri et al. (2008). Para avaliação do método proposto foram utilizadas as 30 instâncias geradas aleatoriamente por Cordeau et al. (2005). Os resultados obtidos por Lopes et al. (2011) são de boa qualidade e em alguns casos as soluções obtidas foram iguais ou melhores que as existentes na literatura até aquele momento.

2.2. SIMULATED ANNEALING – SA

O SA proposto por Kirkpatrick et al. (1983), é uma meta-heurística. A partir de uma solução inicial o algoritmo dá início a uma busca. Em seu procedimento principal, que consiste em um *loop,* são geradas soluções vizinhas de forma aleatória (MAURI et al., 2008). Kirkpatrick et al. (1983) realizaram um estudo intensivo aplicando essa técnica ao Problema do Caixeiro Viajante, obtendo bons resultados.

O SA consiste em um método de busca local que aceita movimentos de piora a fim de escapar de ótimos locais. A técnica inicia sua busca a partir de uma solução inicial qualquer e seu procedimento principal consiste em um *loop* que gera a cada iteração, aleatoriamente, um único vizinho S' da solução S atual (MAURI et al. 2008).

A Figura 4 apresenta um pseudocódigo para o algoritmo SA em um contexto de minimização. Note que a cada geração de um vizinho S' de S, é testada a variação do valor da função objetivo f, ou seja, $\Delta = f(S') - f(S)$. Em um problema de minimização, se $\Delta < 0$ o método aceita a solução e S' passa a ser a nova solução atual. Caso $\Delta \geq 0$, a solução vizinha candidata também poderá ser aceita, entretanto, com uma probabilidade $e^{-\Delta/T}$, onde T é um parâmetro do método chamado de temperatura, usado para regular a probabilidade de aceitação de soluções piores.

```
Simulated Annealing (β, SAmax, T0, Tc e S)
1.   S* ← S;
2.   IterT ← 0;
3.   T ← T0;
4.   enquanto (T > Tc) faça
5.       enquanto (IterT < SAmax) faça
6.           IterT ← IterT + 1;
7.           gerar (um vizinho qualquer S' ∈ ψ(S));
8.           Δ ← f(S') − f(S);
9.           se (Δ < 0)      S ← S';
10.              se (f(S') < f(S*))      S* ← S';      fim-se;
11.          senão
12.              tomar (x ∈ [0,1]);
13.              se ( x < e^(−Δ/T))      S ← S';      fim-se;
14.          fim-se;
15.      fim-enquanto;
16.      T ← β * T;   IterT ← 0;
17.  fim-enquanto;
18.  S ← S*;
19.  retorne S;
fim;
```

Figura 4 - Pseudocódigo genérico da meta-heurística Simulated Annealing (MAURI, 2008).

Inicialmente na linha 3 do pseudocódigo apresentado na Figura 4 a temperatura T assume um valor elevado T_0 e enquanto T não atingir a temperatura de congelamento

T_c, são realizadas um número fixo de iterações *SAmax* (representando o número de iterações para se atingir o equilíbrio térmico em uma dada temperatura), como podemos observar a partir da linha 5. Podemos observar na linha 16 que a temperatura é diminuída de forma gradativa por uma taxa de resfriamento β, sendo $0 < \beta < 1$ e em um dado instante t, $T_t \leftarrow \beta * T_{t-1}$. A solução obtida quando o sistema se encontra estável evidencia um mínimo local, uma vez que soluções piores não são mais aceitas, o que em alguns casos também pode representar um mínimo global (MAURI, 2008).

Mauri (2008) fez uso do SA para o *Dial-a-Ride Problem* (DARP) também conhecido como *Problema de Roteirização e Programação de Veículos.* O DARP consiste em desenvolver rotas de veículos e escalas de funcionários para o transporte de diversos usuários, que por sua vez especificam as requisições de embarque e desembarque entre locais de origem e destino específicos. Esse processo tem como objetivo o planejamento de um conjunto de rotas para veículos que seja capaz de contemplar o maior número de passageiros, obedecendo um conjunto de restrições e visando o custo mínimo. Os autores ainda descrevem que a escolha pelo SA se deu pelos excelentes resultados obtidos para diversos problemas de otimização existentes na literatura, bem como a simplicidade do algoritmo. Os resultados obtidos em todos os casos abordados foram soluções válidas e demandaram pouco tempo de processamento.

Problemas de programação de horários estão presentes em diversas organizações. Santos (2016) propôs um método automatizado para a programação de horários de funcionários de uma loja varejista utilizando o SA para a solução do problema. Era necessário que fossem atendidas as leis trabalhistas, preferências dos funcionários e normas internas da loja. Os resultados obtidos foram confrontados com os resultados de um método exato utilizado previamente para a resolução do problema. O SA se mostrou muito eficiente por convergir em poucos segundos para a solução ótima encontrada na literatura e que o método exato demorou aproximadamente 15 minutos para obtê-la.

2.3. GREEDY RANDOMIZED ADAPTIVE SEARCH PROCEDURE - GRASP

O GRASP, proposto por Feo e Resende (1995) é uma meta-heurística probabilística e iterativa, sendo que em cada iteração é obtida uma solução para o problema e há um reinício. O GRASP contempla duas fases: na primeira é construída uma solução, já na segunda é realizada uma busca local que objetiva melhorar a qualidade da solução construída na primeira fase (FEO e RESENDE, 1995).

Na Figura 5 são apresentadas as etapas do método GRASP para um problema de minimização da função objetivo f.

```
GRASP(α)
1.   f_min ← +∞;
2.   enquanto não CritérioParada faça
3.       S1 ← Construção_Solução(α);
4.       S2 ← Busca_Local(S1);
5.       se f(S2) < f_min então
6.           S ← S2;
7.           f_min ← f(S2);
8.       fim-se
9.   fim-enquanto;
10.  retorne S;
fim;
```

Figura 5 - Pseudocódigo genérico da meta-heurística GRASP.

O GRASP é um método heurístico que faz uso de boas características dos algoritmos ditos puramente gulosos e dos procedimentos aleatórios na fase de construção de soluções viáveis. A cada iteração, a função objetivo f é calculada, e é retornada como resultado a solução que detiver o menor valor.

Podemos observar na Figura 5 que os únicos parâmetros a serem definidos são o parâmetro de aleatoriedade (α) e o critério de parada. Na maioria das aplicações o critério de parada é baseado no número máximo de iterações. Outros critérios também podem ser definidos, por exemplo, estabelecer um limite de tempo em que o procedimento poderá ficar sem encontrar melhores soluções, ou ainda, estabelecer um tempo máximo de execução (RANGEL et al., 2000).

Ainda na Figura 5, note que o GRASP, enquanto o critério de parada verificado na linha 2 do pseudocódigo não for satisfeito, constrói repetidamente uma solução *s*1 (linha 3) e esta solução é melhorada por um procedimento de busca local (linha 4). A

melhor solução obtida até o momento (linhas 5 a 7) é sempre armazenada e ao final do procedimento ela será definida como resultado do método (linha 10). O parâmetro α é visto como o componente probabilístico do GRASP, assim $0 \leq \alpha \leq 1$ e para valores pequenos de α, as soluções construídas são mais gulosas enquanto que para valores maiores de α, as soluções são obtidas de forma mais aleatória (KAMPKE, 2010). Observe que o parâmetro α de entrada do GRASP é utilizado na linha 3 para construção de soluções.

Kampke (2010) propôs um algoritmo que se baseou na meta-heurística GRASP para o problema de sequenciamento de tarefas em máquinas paralelas, com tempos de preparação das máquinas dependentes da sequência e do número de recursos utilizados. Nesse trabalho, o parâmetro de aleatoriedade adotado para a fase de construção do GRASP, se ajusta automaticamente de acordo com a soluções obtidas previamente. Os resultados obtidos foram satisfatórios, somado a isso houve uma melhoria média de 9,14%, quando comparados aos resultados presentes na literatura.

Pinto et al. (2006) propôs a utilização do GRASP para o Problema da Cadeia de Caracteres mais Próxima (PCCP), em que dada uma determinada métrica, é desejado que se obtenha uma sequência de caracteres, pertencente à um conjunto de cadeias de caracteres de mesmo tamanho, que mais se aproxime dessa métrica. Dessa forma, desejou-se minimizar a maior distância desta cadeia de caracteres às demais cadeias presentes no conjunto. Os autores utilizaram quatro versões da meta-heurística GRASP, sendo que todas utilizaram a mesma estratégia para a fase de construção de soluções. Já na fase de busca local foram utilizadas abordagens determinísticas e aleatórias, somadas as estratégias da Primeira Melhoria - *First Improvement* (FI) e Melhor Melhoria – *Best Improvement* (BI). Os resultados encontrados se demonstraram eficientes na resolução do problema e foi possível verificar a robustez do algoritmo, haja vista que os resultados apresentaram uma diferença percentual, na maioria dos casos, inferior a 2%.

3. METODOLOGIA

Nesta seção será apresentada a metodologia abordada neste trabalho.

3.1. FORMULAÇÃO MATEMÁTICA

Assim como apresentado em Cordeau et al. (2005), no que se refere as restrições espaciais, neste trabalho o problema foi tratado em sua forma discreta, onde os berços possuem capacidades de atendimento específicas e contemplam um navio por atendimento. Já do ponto de vista das restrições temporais, o problema foi tratado em sua forma dinâmica, ou seja, os navios possuem datas de chegadas e partidas diferentes. Neste caso, o objetivo principal é a minimização do tempo total gasto pelos navios dentro do porto, que conforme descrito por Hansen et al. (2008), é uma função objetivo apropriada para o PAB.

Assim, é possível descrever algumas variáveis temporais para o problema, como apresentado na Figura 6, considerando um navio i e um berço k (MAURI et al., 2008).

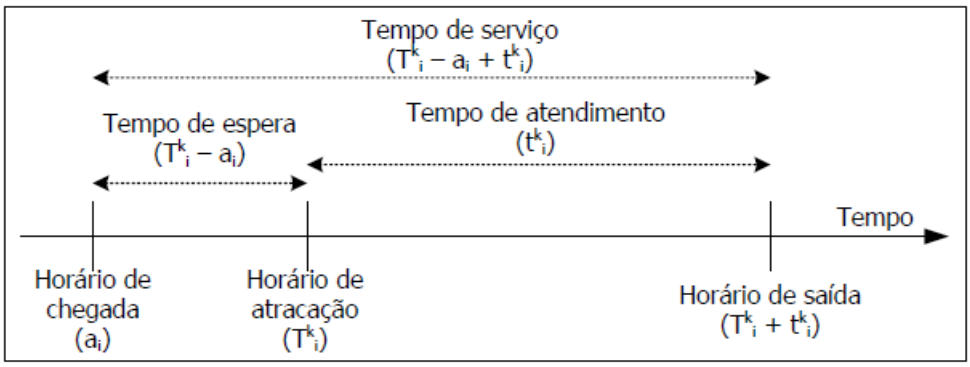

Figura 6 - Variáveis referentes ao tempo (MAURI, 2008).

Neste trabalho, o PAB é abordado com um modelo adaptado do proposto por Cordeau et al. (2005). Dessa forma, considera-se a existência de um conjunto N contendo $|N|$ navios programados para serem atendidos por um conjunto K contendo $|K|$ berços. Considerando-se as restrições espaciais, bem como a existência de restrições operacionais de cada berço, o tempo de atendimento de cada navio em cada berço pode ser diferente, caso o navio i $(i \in N)$ possa ser atendido no berço k $(k \in K)$. Logo, o tempo de atendimento do navio i no berço k é dado como t_i^k e ele é inicialmente conhecido, assim como o tempo previsto de chegada (a_i) de cada navio i ao porto.

Assim, deve-se realizar um planejamento em que seja determinado qual berço k cada navio i irá atracar e o respectivo tempo de atracação T_i^k. Caso um navio chegue ao porto e não haja berço disponível para atendê-lo, ele deverá permanecer aguardando (tempo de espera) até que haja a possibilidade de atracar para ser atendido. Cada navio i possui um tempo limite b_i para sua partida do porto, ou seja, o serviço deve obrigatoriamente ser concluído até esse tempo. Há ainda uma janela de tempo de funcionamento para cada berço k, definido pelo tempo de abertura o^k e de fechamento c^k. Os navios devem ser atendidos dentro desse intervalo.

Dessa forma, haja vista que o objetivo é minimizar o tempo de serviço de cada navio, uma solução S para o PAB possui a seguinte função objetivo:

Minimizar:

$$f(S) = \sum_{i \in N} \left(T_i^k - a_i + t_i^k\right) \text{ se } i \text{ atracou em } k$$

Para que uma solução seja considerada viável, deve-se considerar a seguintes restrições:

- Todos os navios devem ser atendidos;
- Cada navio i deve ser atendido por um único berço k;
- Um berço k não pode atender mais de um navio ao mesmo tempo;
- Cada navio i só pode atracar após a sua chegada no porto ($T_i^k \geq a_i$);
- O atendimento de cada navio i deve terminar antes do seu tempo limite ($T_i^k + t_i^k \leq b_i$);
- Todos os navios devem ser atendidos no período de funcionamento dos berços, ou seja, $o^k \leq T_i^k$ e $T_i^k + t_i^k \leq c^k$, $\forall\ i \in N$ atracado em k.

3.2. REPRESENTAÇÃO DE UMA SOLUÇÃO

A representação da solução adotada neste trabalho é similar a utilizada por Mauri (2008). Dado que $|K|$ e $|N|$ indicam no PAB, respectivamente, a quantidade de berços e navios, uma solução é representada por uma matriz de $|K|$ linhas por $|N|$ colunas.

Dessa forma, cada linha da matriz faz referência a um berço e, por sua vez, as colunas indicam a sequência de atendimento dos navios alocados em cada berço. As demais colunas da matriz, após a coluna contendo o último navio, são preenchidas com o valor -1, para facilitar o conhecimento da quantidade de navios atendidos em cada berço.

A Figura 7 ilustra uma solução para o PAB conforme descrito no parágrafo anterior, em que no berço “1” será atendido somente o navio “3” e no berço “2” serão atendidos os navios “2” e “1”, exatamente nessa ordem. Vale ressaltar que a solução deve atender todas as restrições apresentadas anteriormente.

1	3	-1	-1
2	2	1	-1

Figura 7 - Representação de uma solução.

3.3. HEURÍSTICA CONSTRUTIVA ALEATÓRIA GULOSA

Para a fase de construção do GRASP, foi desenvolvida uma heurística construtiva gulosa aleatorizada.

O algoritmo inicialmente cria uma lista L com todos os navios e ordena crescentemente L pelos tempos de chegada (a_i) de cada navio i ao porto, ou seja, do primeiro ao último navio a chegar. Uma vez que os navios estão ordenados pelo tempo de chegada na lista L, são realizadas iterações sobre ela, em que a cada iteração o primeiro navio de L é removido e inserido na solução S. Para escolher o berço, a ser inserido o navio i, é calculado para cada berço $(k \in K)$ o tempo de início de atendimento do navio (u_i^k), caso seja alocado naquele berço. Os berços são então ordenados crescentemente por u_i^k e é escolhido aleatoriamente um berço dentre os α% berços mais bem ranqueados. Por fim, o navio é inserido na solução S no berço escolhido.

A Figura 8 apresenta um pseudocódigo para o algoritmo de construção heurística utilizado neste trabalho. Como podemos observar, dado um valor α, o algoritmo inicializa n e k com valor 0 (linhas 1 e 2) e executa um procedimento de ordenação dos $|N|$ navios em ordem crescente pelo tempo de chegada a_i armazenando-os na

lista L (linha 4). Em seguida, na linha 6, remove o primeiro navio de L e o atribui a i, então itera através dos $|K|$ berços. Se o berço k não é capaz de atender o navio i, então atribui valor muito grande a u_i^k, como podemos observar nas linhas 8 e 9. Caso contrário, é verificado se o berço está vazio, o que significa que o navio pode ser atendido naquele berço assim que chegar ao porto, e o tempo de chegada a_i é atribuído como u_i^k (linhas 11 e 12). Caso o berço não esteja vazio, então é buscado o término do atendimento do último navio atendido no berço k e é verificado se esse valor é maior que o tempo de chegada a_i, atribuindo-o a u_i^k em caso verdadeiro (linhas 15 a 17). O caso contrário, significa que o navio chegou e o berço está disponível para iniciar o atendimento naquele instante a_i (linha 19).

Por fim, o valor u_i^k é inserido na lista B. Então, os valores em B são ordenados em ordem crescente e os melhores berços são ranqueados com base em α. A partir disso, um deles é escolhido aleatoriamente para alocar o navio na solução S (linhas 24 a 28). Esse procedimento se repete até que sejam estabelecidos os berços de atendimento para cada navio, e por fim, a solução S é retornada (linha 30).

```
Construção_Solução(α)
1.   n ← 0;
2.   k ← 0;
3.   B ← ∅;
4.   L ← listaNaviosOrdenadosTempoChegada(N);
5.   enquanto (n < |N|) faça
6.       i ← removerPrimeiroNavio(L);
7.       enquanto (k < |K|) faça
8.          se (berco_nao_atende(i,k)) então
9.              u_i^k ← ∞;
10.         senão
11.              se (berco_vazio(k)) então
12.                 u_i^k ← a_i;
13.             fim-se
14.             senao
15.                C ← terminoAtendimentoUltimoNavio(k);
16.                se (a_i < C) então
17.                   u_i^k ← C;
18.                fim-se
19.                senão u_i^k ← a_i;
20.         fim-se
21.         B ← B ∪ {u_i^k};
22.         k ← k + 1;
23.      fim-enquanto;
24.      ordenarBercosCrescentemente(pelos valores de u_i^k em B);
25.      melhoresBercos ← α * |B|;
26.      b ← escolher aleatoriamente entre 1 e melhoresBercos;
27.      colocarNavioNoBercoSolucao(i, b, S);
28.      n ← n + 1;
29.  fim-enquanto;
30.  retorne S;
fim;
```

Figura 8 - Pseudocódigo da heurística construtiva aleatória gulosa.

3.4. ESTRUTURA DE VIZINHANÇA

Neste trabalho, para a etapa de geração de soluções vizinhas na meta-heurística SA foram utilizadas três estratégias descritas na literatura.

Foram implementadas as abordagens também utilizadas por Mauri (2008), que consistem, basicamente, em três diferentes movimentos de troca: reordenar navios, realocar navio e trocar navios. Assim como Mauri (2008), esses movimentos garantem que cada navio seja atribuído apenas a berços que sejam capazes de atendê-los. Após a execução de cada um desses movimentos é recalculado o valor da função objetivo para a nova solução.

A Figura 9 mostra o movimento de reordenação de navios, em que basicamente é selecionado um berço aleatoriamente e dentre os navios atendidos por ele é selecionado um navio aleatoriamente, em seguida uma nova posição para ele é selecionada e por fim ocorre a troca de posição entre os navios. Assim é alterada a sequência de atendimento dos navios no berço selecionado.

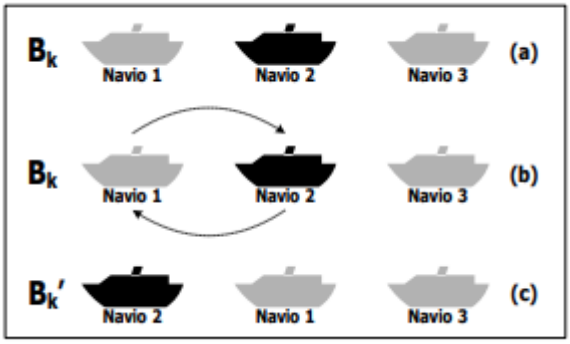

Figura 9 - Movimento reordenar navios. (MAURI, 2008)

A Figura 10 ilustra o movimento de realocação de navios, em que basicamente são selecionados dois berços aleatoriamente e um navio aleatoriamente dentre os dois berços, e por fim o navio é tirado de seu berço atual e atribuído ao outro. Nesse movimento, o novo berço selecionado deverá ter a capacidade de atender o navio, senão outro berço será selecionado. Por fim a sequência de atendimento no berço é reordenada pelo horário de chegada dos navios.

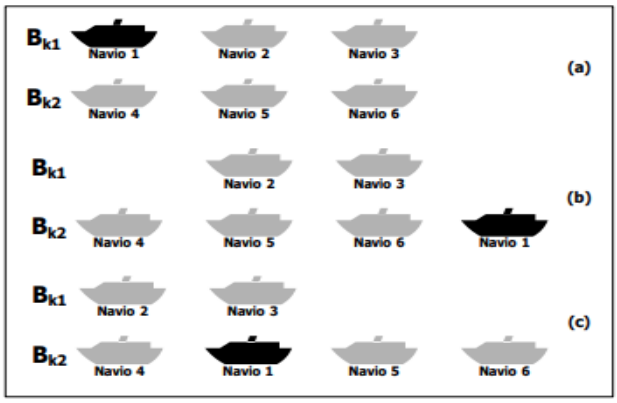

Figura 10 - Movimento realocar navios. (MAURI, 2008)

Por fim, o terceiro movimento é o de troca de navios, apresentado na Figura 11, que consiste basicamente na seleção de dois navios em berços diferentes que serão trocados para as respectivas posições em que um e o outro se encontravam. Caso não haja a possibilidade de atendimento dos navios nos novos berços, são selecionados novos navios e/ou berços. Após a troca a sequência de atendimento é reordenada pelo horário de chegada dos navios.

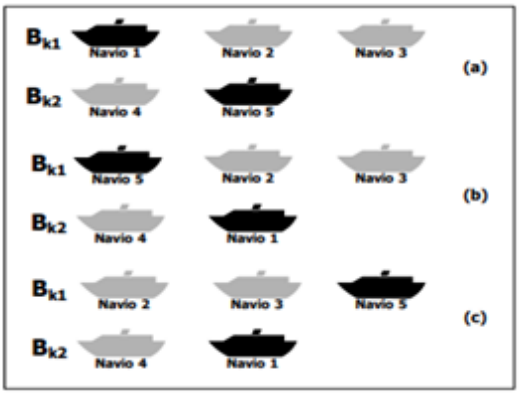

Figura 11 - Movimento trocar navios. (MAURI, 2008)

3.5. GRASP+SA PARA O PAB

Nesse trabalho foi implementada a meta-heurística GRASP somado ao SA como algoritmo para a fase de busca local.

Note que o algoritmo proposto neste trabalho, mostrado na Figura 12, segue a mesma estrutura apresentada no pseudocódigo da Figura 6, entretanto, na fase de busca local (linha 4) é utilizado um procedimento que implementa o SA (apresentado na seção

2.3.1), fornecendo como parâmetro de entrada a solução S_1, obtida na primeira fase do algoritmo GRASP.

```
GRASP(α)
    f_min ← +∞;
    enquanto não CritérioParada faça
        S_1 ← Construção_Solução(α);
        S_2 ← Simulated_Annealing(β, T_0, T_c, SAmax e S_1);
        se f(S_2) < f_min então
            S ← S_2;
            f_min ← f(S_2);
        fim-se
    fim-enquanto;
    retorne S;
fim;
```

Figura 12 - Pseudocódigo da meta-heurística proposta GRASP+SA.

4. RESULTADOS

Nesta seção são apresentados e discutidos os resultados obtidos neste trabalho.

4.1. INSTÂNCIAS PARA O PAB

Para avaliar o desempenho do método proposto neste trabalho foram utilizadas cinco instâncias distintas, cada uma contendo 60 navios e 13 berços. Esse conjunto representa apenas 1/6 das instâncias utilizadas por Mauri (2008), que descreveu o processo de geração delas por Cordeau et al. (2005) como aleatório.

Conforme Mauri (2008), um fato que dificulta a comparação direta com métodos já propostos na literatura é que geralmente as empresas que cedem os dados para essas instâncias não permitem a divulgação deles. Assim, as instâncias utilizadas neste trabalho foram obtidas através do Prof. Dr. Geraldo Regis Mauri durante o curso da disciplina de Métodos de Otimização ofertada pelo Departamento de Computação do CCENS - UFES.

A escolha dessas instâncias ocorreu pelo prévio acesso a elas, a dificuldade em obter instâncias baseadas em informações reais dos portos, e o conhecimento dos valores ótimos para fins de comparação e análise. Conhecer os valores ótimos para cada uma delas contribuiu para análises apresentadas mais adiante.

4.2. PARÂMETROS PARA O GRASP

O valor de α (parâmetro de entrada para o GRASP) foi definido empiricamente por meio de testes. Já o critério de parada foi definido em termos do tempo de execução do algoritmo, assim o tempo máximo para cada execução foi estabelecido em 300 segundos.

Basicamente foram testados diferentes valores de α. De forma mais específica, foram testados valores de α no conjunto: {0; 0,2; 0,5; 0,8; 1}. Para cada instância, foram realizadas cinco execuções com cada valor candidato de α, totalizando 125 execuções.

Ao calcularmos o valor da função objetivo para cada solução construída, observou-se que para α igual a 0 era construída a mesma solução, o que já era esperado, e assim obtinha-se o mesmo valor de f, pois o algoritmo sempre escolhia o berço cujo tempo

de início de atendimento ao navio era o menor. Dessa forma, foram geradas soluções viáveis, entretanto não havia a possibilidade da geração de soluções melhores, uma vez que o algoritmo para esse valor, não era capaz de escolher um berço que demorasse mais tempo para iniciar o atendimento ao navio, escolha que talvez permitiria piora ou melhora na construção da solução.

Para valores de α maiores que 0.2, observou-se que as soluções construídas em geral foram piores que as soluções obtidas com α igual a 0, muitas delas inviáveis, pois eram escolhidos aleatoriamente berços que não eram capazes de atender o navio, sendo penalizadas durante o cálculo de f.

Por fim, para α igual a 0.2, em média foram obtidas soluções melhores, quando comparadas às construídas com os demais valores de α, e isso se deve a possibilidade de escolha entre os 20% melhores tempos de início de atendimento ao navio. Devido os resultados obtidos nos testes com esse valor serem considerados viáveis e em média despontarem em relação aos demais, ele foi estabelecido como valor para o parâmetro α.

4.3. PARÂMETROS PARA O SA

Os parâmetros utilizados pelo SA em todos os testes foram: SAmax que representa o número máximo de iterações para atingir o equilíbrio térmico, definido em $\frac{|K| \times |N|}{2}$, que neste trabalho equivale a 390, o valor T_c que representa a temperatura de congelamento, fixado em 0.01, a solução S construída através da heurística construtiva aleatória, e finalmente, de forma similar ao processo de definição do parâmetro α, os parâmetros β e T_0, foram definidos empiricamente por meio de testes.

Basicamente foram testados diferentes valores para β e T_0. Especificamente foram testados valores de β no conjunto: {0,999; 0,995; 0,975} e T_0 no conjunto: {500; 750; 1000}. Para cada instância, foram realizadas cinco execuções para cada combinação possível entre os valores de cada conjunto, ou seja, cinco execuções para cada instância em cada uma das nove combinações de valores. A cada execução do algoritmo, a escolha do movimento para gerar uma solução vizinha, deu-se de forma aleatória entre os três movimentos apresentados na metodologia deste trabalho.

Ao analisar os valores de f encontrados em cada execução, observou-se que independentemente dos valores de β e T_0 escolhidos como entrada, o SA obteve sempre os mesmos cinco valores, para cada instância. Assim, a média e o desvio padrão encontradas eram sempre o mesmo, entretanto o tempo decorrido para encontrar a melhor solução foi diferente.

Analisando o tempo decorrido para encontrar a melhor solução, observou-se que para β=0.975 e T_0=750, esse tempo foi, em média, menor quando comparado aos demais valores. Dessa forma, foram estabelecidos β=0.975 e T_0=750.

4.4. AMBIENTE DE TESTES

Todos os experimentos apresentados neste trabalho foram realizados em um computador com processador Intel Core i7 (11° Geração) de 3GHz com 4 núcleos e 8 threads, sistema operacional Windows 11 e 32GB de memória RAM. Toda a implementação foi desenvolvida na linguagem de programação C++.

4.5. RESULTADOS COMPUTACIONAIS

Após realizar a definição dos valores dos parâmetros de entrada para o GRASP + SA, foram realizadas 10 execuções em cada instância.

Em seguida, foram analisadas as soluções obtidas dessas execuções, e notou-se que os valores de f eram sempre os mesmos e todas as restrições eram atendidas, ou seja, a quantidade de violações das restrições foi nula. A Tabela 1, resume essas informações.

Instâncias	f_{best}	f_{avg}	f_{sd} (%)
i01	1409	1409	0
i02	1368	1368	0
i03	1376	1376	0
i04	1359	1359	0
i05	1344	1344	0

Tabela 1 – Resultados obtidos.

Na Tabela 1 a coluna *Instâncias* identifica cada instância utilizada e a coluna f_{best} representa o melhor valor obtido para a instância dentre as 10 execuções realizadas.

Por fim, a coluna f_{avg} apresenta a média entre os 10 valores de f obtidos e a coluna f_{sd} mostra o valor do desvio padrão entre eles.

Note que os resultados dos testes apresentados na Tabela 1, indicam a consistência e robustez da combinação das meta-heurística GRASP e SA, que em todas a execuções encontraram os mesmos valores de f.

Na Tabela 2 podemos observar a qualidade e velocidade para obtenção das soluções.

Instâncias	f^*	f_{best}	Δ **(%)**	ω **(seg)**
i01	1409	1409	0	0,3176
i02	1368	1368	0	0,3160
i03	1376	1376	0	0,3285
i04	1359	1359	0	0,3205
i05	1344	1344	0	0,3227

Tabela 2 - Comparação de resultados.

Assim como na Tabela 1, a coluna *Instâncias* na Tabela 2, identifica cada instância, já a coluna f^* representa o valor ótimo real conhecido para a instância, a coluna f_{best}, são os mesmos valores apresentados na Tabela 1. A coluna Δ, contém a diferença percentual entre as colunas f^* e f_{best} e por fim, a coluna ω, indica o menor tempo para se obter f_{best}.

Observando a coluna Δ, podemos concluir que a meta-heurística hibrida GRASP+SA foi capaz de obter boas soluções e de forma rápida, como apresentado na coluna ω. Note que o valor de Δ para todas as instâncias foi nulo, indicando que a meta-heurística híbrida proposta neste trabalho foi capaz de obter o valor ótimo real. Isso se deve principalmente a etapa de intensificação de busca, ou seja, ao SA, que sempre que executado realiza tentativas de melhoria da solução passada a ele como parâmetro após a construção realizada na primeira fase do GRASP.

5. CONCLUSÕES

Neste trabalho foi tratado o Problema de Alocação de Berços (PAB) sendo abordado com um modelo adaptado do proposto por Cordeau et al. (2005). No que se refere as restrições espaciais o problema foi tratado em sua forma discreta, já em relação as restrições temporais em sua forma dinâmica, sendo o objetivo principal a minimização do tempo total (tempo de serviço) gasto pelos navios dentro do porto.

Para a solução do PAB foi utilizada a meta-heurística híbrida GRASP+SA, ou seja, a combinação das meta-heurísticas, sendo o *Simulated Annealing* usado como algoritmo de intensificação de busca.

Os resultados obtidos mostraram que a aplicação da meta-heurística híbrida GRAP+SA para solucionar o PAB foi eficiente. A etapa de intensificação de busca foi capaz de melhorar as soluções obtidas através da heurística construtiva aleatória utilizada para fase de construção do GRASP e atender todas as restrições do problema. Além disso, soluções foram geradas em um tempo computacional muito baixo, sendo necessário menos de um segundo para se atingir o valor ótimo nas instâncias usadas nos experimentos computacionais.

Para trabalhos futuros, recomenda-se o uso do GRASP combinado com outras meta-heurísticas na etapa de busca local, como o VNS.

REFERÊNCIAS BIBLIOGRÁFICAS

BIERWIRTH, C. & MEISEL, F., (2010), **A survey of berth allocation and quay crane scheduling problems in container terminals**, European Journal of Operational Research, 202, issue 3, p. 615-627.

BROWN, G.G., LAWPHONGPANICH, S. & THURMAN, K.P. (1994). **Optimizing ship berthing.** Naval Research Logistics, 41, 1-15.

BROWN, G.G., CORMICAN, K.J., LAWPHONGPANICH, S. & WIDDIS, D.B. (1997). **Optimizing submarine berthing with a persistence incentive.** Naval Research Logistics, 44, 301-318.

CORDEAU, J.-F., LAPORTE, G., LEGATO, P., & MOCCIA, L. (2005). **Models and tabu search heuristics for the berth-allocation problem.** Transportation Science, 39(4), 526-538.

FEO, T.A. & RESENDE, M.G.C. (1989). **A probabilistic heuristic for a computationally difficult set covering problem.** Operations Research Letters, 8, 67-71.

HANSEN, P.; OĞUZ, C. & MLADENOVIĆ, N. (2008). **Variable neighborhood search for minimum cost berth allocation.** European Journal of Operational Research, 191, 636-649.

IMAI, A.; NISHIMURA, E. & PAPADIMITRIOU, S. (2001). **The dynamic berth allocation problem for a container port.** Transportation Research Part B: Methodological, v. 35, n. 4, p. 401-417.

IMAI, A., NISHIMURA, E. & PAPADIMITRIOU, S. (2003). **Berth allocation with service priority**. Transportation Research Part B, 37, 437-457.

KAMPKE, E. H. (2010). **Meta-heurísticas para o Problema de Programação de Tarefas em Máquinas Paralelas com Tempo de Preparação Dependentes da Sequência e de Recursos.** Dissertação de Mestrado – Universidade Federal de Viçosa (UFV), Viçosa.

KIRKPATRICK, S.; GELATT JR C. D. & VECCHI, M.P. (1983). **Optimization by Simulated Annealing**. Science, 220, 4598, 671-680.

LOPES, A. T.; SCHULZ, V. M. L. & MAURI, G. R. **Grasp Com Path Relinking Para O Problema De Alocação De Berços**. Pesquisa Operacional para o Desenvolvimento, [S. l.], v. 3, n. 3, p. 218–229, 2011.

MAURI, G. R. **Novas abordagens para representação e obtenção de limitantes e soluções para alguns problemas de otimização combinatória**. Tese (Doutorado em Computação Aplicada) - Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2008.

MAURI, G. R., OLIVEIRA, A. C. M. & LORENA, L. A. N. **Resolução do problema de alocação de berços através de uma técnica de geração de colunas**. Pesquisa Operacional [online]. 2010, v. 30, n. 3, pp. 547-562.

PINTO, E. R.; CABRAL, L. A. F. & MACAMBIRA, E. M. **Meta-heurística GRASP Na Resolução Do Problema Da Cadeia De Caracteres Mais Próxima.** SIMPÓSIO BRASILEIRO PESQUISA OPERACIONAL, 38., 2006, Goiânia.

RANGEL, M. C.; ABREU, N. M. M. & BOAVENTURA-NETTO, P.O. **Grasp para o pqa: um limite de aceitação para soluções iniciais.** Pesquisa Operacional**,** 20(1)**:**45-58, 2000.

RODRIGUES, I. B. G.; ROSA, R. A.; GOMES T.C. & RIBEIRO G.M. **Modelo matemático para o Problema de Alocação de Berços em portos com limitações de operação de carga ao longo do cais.** Gestão & Produção [online]. 2016, v. 23, n. 04.

SANTOS, I. C. V. **Utilização Da Meta-heurística Simulated Annealing Para a Otimização Da Programação De Turnos Dos Funcionários De Uma Loja Varejista.** TCC – Universidade Federal de Ouro Preto (UFOP), Ouro Preto, 2016.

THURMAN, K.P. (1989). **Optimal ship berthing plans. Dissertation (Masters of Science in Operations Research)**, Naval Postgraduate School - USA.

ZHENHUB. **Cargo Ship Delays in 2022**. 2022. Disponível em: <https://zhenhub.com/blog/cargo-ship-delays-2022/#understanding-the-causes-of-cargo-ship-delays>. Acesso em: 05 out. 2022.