Universidade Federal do Rio Grande - FURG
Centro de Ciências Computacionais
Escola de Engenharia
Instituto de Matemática, Estatística e Física
Programa de Pós Graduação em Modelagem Computacional

Elizangela Dias Pereira

O Problema de Alocação de Berços: Um estudo das heurísticas Simulated Annealing e Algoritmo Genético

Rio Grande - RS
2013

ELIZANGELA DIAS PEREIRA

O PROBLEMA DE ALOCAÇÃO DE BERÇOS: UM ESTUDO DAS HEURÍSTICAS SIMULATED ANNEALING E ALGORITMO GENÉTICO

Dissertação apresentada ao Programa de Pós Graduação em Modelagem Computacional da Universidade Federal do Rio Grande, como requisito parcial para obtenção do Grau de Mestre. Área de Concentração: Modelagem Computacional.

Orientadora: Profª. Drª. CATIA MARIA DOS SANTOS MACHADO
Co-orientador: Prof. Dr. MILTON LUIZ PAIVA DE LIMA

Rio Grande - RS
2013

ELIZANGELA DIAS PEREIRA

O PROBLEMA DE ALOCAÇÃO DE BERÇOS: UM ESTUDO DAS HEURÍSTICAS SIMULATED ANNEALING E ALGORITMO GENÉTICO

Dissertação apresentada ao Programa de Pós Graduação em Modelagem Computacional da Universidade Federal do Rio Grande, como requisito parcial para obtenção do Grau de Mestre. Área de Concentração: Modelagem Computacional.

Aprovada em 04 de fevereiro de 2013.

BANCA EXAMINADORA

Cotp aprovado.

Profa. Dra. CATIA MARIA DOS SANTOS MACHADO
Orientadora - FURG

Prof. Dr. MILTON LUIZ PAIVA DE LIMA
Co-Orientador - FURG

Prof. Dr. ALESSANDRO DE LIMA BICHO
FURG

Andréa C. Konrath

Profa. Dra. ANDREA CRISTINA KONRATH
UFSC

Rio Grande-RS
2013

Para meus pais *Helen* e *Nilza*, por todo amor e dedicação.

Agradecimentos

Primeiramente agradeço a Deus por estar sempre ao meu lado, Nele sempre encontro a força que preciso.

Agradeço aos meus pais, Helen e Nilza por simplesmente tudo. Aos meus irmãos Luiz e Mário e suas famílias, que contribuem, cada um do seu jeito para que eu continue crescendo.

Agradeço a presença constante, mesmo que distante dos meus melhores amigos: Daniel, Rita e Carolina. Obrigada por fazerem parte da minha história, por aceitarem tripular ao meu lado nessa viagem maravilhosa e por terem me mantido firme e forte no caminho.

Ao Tiago Klug pelo desenvolvimento do software e à Merhy Heli que dividiu comigo bons momentos na busca pelo conhecimento.

Às professoras Bárbara Denicol, Cristiana Poffal e Cinthya Schneider pela oportunidade como bolsista no projeto, propiciando o contato e aprendizado de LaTeX.

À professora Catia pela orientação neste trabalho, pela paciência e dedicação, e principalmente pelo laço de amizade estabelecido no decorrer deste trabalho.

Ao professor Milton pela co-orientação e carinho.

A FURG e CAPES pelo apoio financeiro. Aos professores da Banca Examinadora.

Resumo

Este trabalho apresenta um estudo de caso das heurísticas *Simulated Annealing* e Algoritmo Genético para um problema de grande relevância encontrado no sistema portuário, o Problema de Alocação em Berços. Esse problema aborda a programação e a alocação de navios às áreas de atracação ao longo de um cais. A modelagem utilizada nesta pesquisa é apresentada por Mauri (2008) [28] que trata do problema como uma Problema de Roteamento de Veículos com Múltiplas Garagens e sem Janelas de Tempo. Foi desenvolvido um ambiente apropriado para testes de simulação, onde o cenário de análise foi constituido a partir de situações reais encontradas na programação de navios de um terminal de contêineres. Os testes computacionais realizados mostram a performance das heurísticas em relação a função objetivo e o tempo computacional, afim de avaliar qual das técnicas apresenta melhores resultados.

Palavras-chave: Problema de Alocação de Berços. Algoritmo Genético. *Simulated Annealing*.

Abstract

This paper presents a case study of heuristics Simulated Annealing and Genetic Algorithm into a problem of great relevance found in the port system, the Berth Allocation Problem. This issue discusses the programming and allocating ships to berthing areas along a quay. The model used in this research is presented by Mauri (2008) [28] that treats the problem as a Vehicle Routing Problem with Multiple Garages and without Time Windows. We developed a testing environment for simulation, where scenario analysis was composed from real situations encountered in scheduling ships a container terminal. The computational tests show the performance of the heuristics with respect to computational time and objective function, to determine which technique is best used.

Keywords: Berth Allocation Problem. Genetic Algorithm. Simulated Annealing.

Sumário

Lista de Abreviaturas

AG	Algoritmo Genético
CX	*Cycle Crossover* (Cruzamento por Ciclo)
FCFS	*First Come First Served* (Primeiro que chega, primeiro atendido)
F.O.	Função Objetivo
HX	*Heuristic Crossover* (Cruzamento Heurístico)
OX	*Order Crossover* (Cruzamento de Ordem)
PAB	Problema de Alocação em Berços (*Berth Allocation Problem*)
PAG	Problema de Atribuição de Guindastes
PL	Programação Linear
PMX	*Partially-Mapped Crossover* (Cruzamento de Mapeamento Parcial)
PO	Pesquisa Operacional
PRV	Problema de Roteamento de Veículos
PRVMG	Problema de Roteamento de Veículos com Múltiplas Garagens
SA	*Simulated Annealing* (Recozimento Simulado)

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A crescente evolução do comércio, bem como do transporte de cargas conteinerizadas no mundo tem forte efeito na necessidade do aumento do tamanho e da quantidade de navios, produzindo assim um impacto muito grande na infra-estrutura dos terminais de contêineres. Dessa forma, buscam-se ferramentas que auxiliem a tomada de decisão de maneira a atender o crescimento da demanda de navios sem que haja aumento de custos, seja por demora no atendimento ou ineficiência da operação.

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 em Berços (*Berth Allocation Problem*). O problema de alocação de berços (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 (onde?) e ao tempo (quando?) em que o navio deverá atracar (Imai et al., 2001)[21].

O PAB é um dos problemas presentes no sistema portuário e que prejudica a logística fundamental para o bom desempenho dos portos, e como a maioria dos berços é operada por companhias de navegação particulares, poucos estudos têm sido conduzidos em relação à alocação de navios em berços, (Imai et al., 2001) [21]. Sendo assim, há uma necessidade crescente de estudos em tecnologias que resultem em ferramentas eficientes para solucionar o PAB.

Pesquisas tem sido desenvolvidas para testes competitivos de novas implementações de algoritmos projetados para evitar mínimos locais, (Rayaward-Smith, 1996) [39]. Esses algoritmos não garantem encontrar uma solução ótima e não é conhecido um limitante de pior caso. Existe um interesse crescente nessa classe de algoritmos pela facilidade de implementação e, principalmente, a sua robustez para resolver uma grande variedade de problemas.

A escolha pelos algoritmos *Simulated Annealing* (SA) e Algoritmo Genético (AG) dá-se pelo fato de que são algoritmos que apresentam vantagens como versatilidade,

eficiência e simplicidade. Além disso, a modelagem do PAB possui características próprias, cujas técnicas heurísticas SA e AG podem sofrer modificações, tornando-se adaptáveis ao problema específico.

1.1 Objetivos

Considerando o problema de alocação em berços (PAB) como um determinante na eficiência operacional de um porto, e sabendo que a elaboração de um plano de atracação, ou seja, a alocação de navios a berços é dispendiosa devido à quantidade de variáveis que acompanham o problema, é de suma importância o uso de uma abordagem computacional.

Deste modo, o objetivo geral do estudo é comparar duas técnicas heurísticas aplicadas ao problema específico de programação e alocação de navios em berços, como um auxílio na tomada de decisão deste contexto em um complexo portuário, propiciando a redução dos custos operacionais.

Para alcançar o objetivo geral, os seguintes objetivos específicos serão considerados no desenvolvimento da pesquisa:

- aplicar um modelo matemático de Programação Linear para resolver um problema construído a partir de situações reais encontradas no setor portuário;
- especificar características do problema PAB a fim de estabelecer estratégias de busca de soluções;
- implementar os algoritmos SA e AG;
- desenvolver um ambiente apropriado para testes de simulação dos algoritmos SA e AG;
- identificar instâncias do problema para as quais a performance do algoritmo SA é diferente ou semelhante à do AG;
- avaliar a qualidade de solução obtida e o tempo de execução dos algoritmos.

Em suma, os objetivos evidenciam a importância da resolução do problema, buscando uma melhoria na logística operacional dos portos, bem como a avaliação de métodos propostos na literatura, visando uma melhora com relação ao tempo computacional e à praticidade operacional.

1.2 Contribuições

Espera-se contribuir com o estudo de métodos de solução para o PAB a partir da proposição de soluções utilizando as técnicas heurísticas SA e AG neste contexto,

visto que estes algoritmos trabalham bem para problemas de otimização combinatória e apresentam vantagens como versatilidade, eficiência e simplicidade. De fato, o crescente interesse nesta classe de algoritmos é atribuída a sua facilidade de implementação e, principalmente, a sua robustez para resolver uma grande variedade de problemas, evidenciando a importância de sua resolução buscando uma melhora na logística operacional dos portos.

Além disso, mostrar que existem métodos eficientes em termos de tempo computacional e praticidade operacional para a resolução do problema específico PAB e, ainda, enriquecer a literatura a respeito do estudo desse problema, visto que é um problema recorrente e cada vez assume uma importância maior no cenário do comércio mundial.

1.3 Organização do Trabalho

Este trabalho está organizado em seis capítulos, incluindo esta introdução. O Capítulo 3 caracteriza o problema de alocação de berços no âmbito do transporte marítimo e sua modelagem como um problema de otimização inserido na área de pesquisa operacional.

No Capítulo 2, faz-se uma revisão sobre conceitos básicos da Pesquisa Operacional, assim como uma descrição de Otimização Combinatória e Heurísticas, conceitos estes necessários para o desenvolvimento da pesquisa. As técnicas heurísticas *Simulated Annealing* e Algoritmo Génetico são apresentadas em detalhe no Capítulo 4.

A análise dos resultados e o aplicativo criado para a aplicação das técnicas heurísticas *Simulated Annealing* e Algoritmo Génetico são descritos no Capítulo 5.

No Capítulo 6, encontram-se as conclusões e as recomendações para trabalhos futuros.

Capítulo 2

Fundamentos Teóricos

Este capítulo tem por objetivo apresentar alguns conceitos preliminares sobre pesquisa operacional e problemas de otimização, bem como sua formulação matemática. Ao final são apresentadas algumas heurísticas de solução para certos problemas de otimização combinatória.

2.1 Modelos em Pesquisa Operacional

A Pesquisa Operacional (PO), como ciência, estrutura processos, propondo alternativas de ação, fazendo assim, a previsão e comparação de valores, de eficiência e de custos. Seus fundamentos encontram-se na Matemática, na Análise de Sistemas e na Estatística. A ferramenta de resolução, por excelência, é o computador, visto que para os problemas reais aos quais as técnicas de PO normalmente se aplicam, acabam conduzindo à construção de modelos matemáticos de médio e grande porte, cuja solução manual é difícil ou mesmo impraticável (Loesch, 1999) [25].

A origem da PO tem a ver com o esforço envolvido nos quase seis anos da 2ª Guerra Mundial. Durante aquela época, foram desenvolvidas experiências que consistiram na criação de grupos integrados por especialistas com conhecimentos diversificados, aos quais foram propostos problemas operacionais bélicos. Esses grupos tiveram sucesso ao oferecer sugestões operacionais criativas e eficazes. Essa modalidade metodológica foi denominada Pesquisa Operacional e, mais tarde, foi trazida para o ambiente civil.

O marco definitivo na afirmação da PO foi a publicação feita por George Dantzig, em 1947, do Método Simplex para a Programação Linear (PL). Assim, a PL tornou-se a primeira técnica explícita, e até hoje a mais fundamental de todas as técnicas da PO.

Em estudos de Pesquisa Operacional, o uso de modelos faz parte de sua própria essência. Trata-se de um recurso adotado para problemas complexos que desafiam a criatividade humana (Pizzolato e Gandolpho, 2009) [37]. Um modelo é uma repre-

sentação da realidade, é através dele que se procura obter os aspectos relevantes de algum problema ou sistema. Os modelos matemáticos constituem de uma importante abstração do mundo real, representado por um conjunto de relações e equações.

Entretanto, existem problemas com alto grau de dificuldade cuja solução muitas vezes não é encontrada facilmente. Nesses casos, há a necessidade da contratação de uma equipe, da mobilização de recursos e o desenvolvimento de um projeto de pesquisa operacional através de estudos detalhados com vistas à construção de um modelo. Esse modelo deve simplificar a realidade, mas preservar as relações do problema real quanto sua causa e efeito.

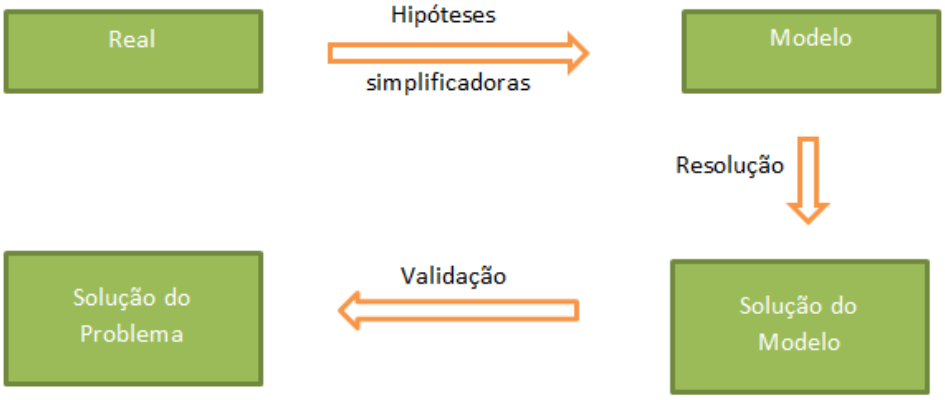


Figura 2.1: *Uso de modelos em PO - adaptado de Pizzolato e Gandolpho [37].*

O procedimento descrito na Figura 2.1 indica basicamente a metodologia adotada para o desenvolvimento de um projeto de PO. Para formular as hipóteses são feitos levantamento de dados, entrevistas, avaliação da natureza do problema, etc., até que se obtenha o modelo. A partir do modelo pode-se escolher as técnicas de resolução aliadas ao uso de recursos computacionais para posterior validação do modelo, se este corresponder a uma solução consistente, caso contrário, devem ser reavaliadas as hipóteses, bem como as técnicas de resolução até que se tenha um modelo que possa descrever a realidade.

Basicamente, os passos para construção de um projeto de PO podem ser decompostos em cinco etapas (Pizzolato e Gandolpho, 2009)[37]:

1. Identificação do problema;
2. Construção do modelo;
3. Determinação da solução do modelo;
4. Teste e validação da solução proposta;
5. Implementação da solução.

2.1.1 Programação Linear

Dentre as técnicas de solução de problemas de PO, destacam-se as técnicas da Programação Linear e suas extensões. A PL constitui-se num modelo simbólico que utiliza a linguagem matemática para descrever as variáveis de decisão. Todo problema de PL pode ser descrito através de uma função objetivo e de um conjunto de restrições, todos lineares. Dessa forma, tem-se o seguinte modelo genérico:

$$
\begin{aligned}
&\{\text{Max, Min}\}\, Z = c_1x_1 + c_2x_2 + \ldots + c_nx_n \\
&\text{sujeito à} \\
&a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n\{=, \leq, \geq\}b_1 \\
&a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n\{=, \leq, \geq\}b_2 \\
&\vdots \\
&a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n\{=, \leq, \geq\}b_m \\
&x_1, x_2, \ldots x_n \geq 0
\end{aligned}
$$

Onde:

$x_1, x_2, \ldots, x_n$ é o conjunto de variáveis estruturais do problema;

$c_1, c_2, \ldots, c_n$ são os coeficientes da função objetivo;

a_{ij} e b_i são os coeficientes das restrições.

A modelagem matemática constitui o primeiro passo na busca da solução de um problema de Programação Linear. Um problema após formulado pode ser resolvido computacionalmente, com o uso de técnicas adequadas, e seus resultados podem ser interpretados e implementados.

Destacam-se como áreas de aplicação da programação linear:

a) Problemas de alocação, ou seja, problemas envolvidos na alocação de recursos escassos entre fins alternativos, de acordo com algum critério.

b) Problemas complexos de alocação que não podem ser resolvidos satisfatoriamente com as técnicas analíticas convencionais.

Alguns exemplos de problemas de alocação:

a) Determinação dos produtos a serem fabricados, levando em consideração a demanda esperada, a adequabilidade e as capacidades da produção e facilidades de distribuição, bem como as diretrizes administrativas, com o objetivo de maximizar os lucros.

b) Problemas de mistura ou combinação de ingredientes utilizados na fabricação dos produtos, tendo em vista a disponibilidade e os custos relativos dos ingredientes, no intuito de encontrar a combinação adequada que resultará no custo mínimo de material por unidade do produto final.

c) Problemas de transporte e distribuição física, ou problemas de *scheduling*, onde pergunta-se qual o plano físico de distribuição que estará tanto dentro das restrições de capacidade como da demanda e que ao mesmo tempo minimize os custos de produção e de distribuição durante o período de planejamento.

2.2 Otimização Combinatória

2.2.1 Introdução

Problemas de otimização consistem em achar a melhor combinação dentre um conjunto de variáveis para maximizar ou minimizar uma função, geralmente chamada de função objetivo ou função custo. Esses problemas podem ser divididos em três categorias: aqueles cujas variáveis assumem valores reais (ou contínuos), aqueles cujas variáveis assumem valores discretos (ou inteiros) e aqueles em que há variáveis inteiras e contínuas, classificados, respectivamente, como problemas de Otimização Contínua, Otimização Combinatória ou Discreta, e Otimização Mista.

Para resolver um problema de otimização combinatória, a ideia mais ingênua é simplesmente combinar, ou seja, enumerar, todas as possíveis soluções, criar todos os subconjuntos existentes a partir do conjunto e das regras de restrição e escolher o de menor custo. No entanto, este procedimento torna-se impraticável a medida que o conjunto inicial cresce. Um exemplo clássico encontrado na literatura é o Problema da Mochila (*Knapsack problem*), que consiste em carregar uma mochila com capacidade limitada, com um conjunto de objetos com pesos e valores diferentes. O objetivo é ocupar a mochila com o maior valor possível, sem extrapolar o seu peso máximo (Marques e Arenales, 2002) [27].

O número de soluções para o Problema da Mochila é dado por:

$$\text{num\_soluções} = \sum_{i=1}^{n-1} C_{n,i} + 1,$$

onde n é o número de elementos no conjunto e C representa a operação matemática de combinação $C_{n,p} = \dfrac{n!}{p!(n-p)!}$.

Dessa forma, quanto maior o número n, mais rapidamente cresce o número de soluções, tornando inviável a enumeração de todas as possibilidades, mesmo utilizando computadores de grande capacidade de processamento. É, portanto, necessário resolver os problemas de otimização combinatória através de outras técnicas, que

forneçam atalhos para a descoberta dos valores ótimos para os problemas propostos ou, pelo menos, boas soluções de cunho geral.

Além do Problema da Mochila existem diversos tipos de problemas de otimização combinatória que podem ser utilizados para o desenvolvimento de sistemas de carga de conteineres, carregamento de caminhões de entrega, investimento de capital e corte, empacotamento, etc. Entre os principais problemas matemáticos de aplicação pratica, é possível citar (Goldarg e Luna, 2000) [18]:

a) problema de particionamento, utilizado na recuperação de informações em bancos de dados, alocação de tripulações em linhas áreas, distribuição do tráfego de comunicações em satélites, alocação de serviços de emergência; roteamento de petroleiros, entre inúmeros outros;

b) problema da árvore geradora mínima: utilizado no projeto de redes de comunicações, roteamento de meios terrestres na defesa da costa, roteamento de veículos com função multiobjetivo (envolvendo, além da distância, pedágios, condições de risco e quaisquer outras variáveis importantes para o modelo);

c) problema de roteamento de veículos: utilizado para definir escalas de tripulação, programação de tarefas e tripulação, rotear veículos com restrições (combustível, janelas de tempo, congestionamento);

d) problemas de fluxo: utilizado principalmente em redes de transporte de energia, contribuindo no planejamento da expansão e distribuição do sistema, telecomunicações e escala de motoristas;

As técnicas clássicas de otimização são confiáveis e possuem aplicações na engenharia e em diversas outras ciências. Porém, estas técnicas apresentam dificuldades numéricas e problemas de robustez, muitas vezes relacionados com a falta de continuidade das funções a serem otimizadas ou de suas restrições, funções não convexas, existência de ruídos nas funções, necessidade de se trabalhar com valores discretos para as variáveis, existência de mínimos ou máximos locais, etc. Dessa forma, os estudos de métodos heurísticos vem sendo conduzidos no sentido de driblar tais dificuldades, principalmente devido ao avanço dos recursos computacionais, pois um fator limitante destes métodos é a necessidade de um número elevado de avaliações da função objetivo, (Schwefel e Taylor, 1994) [43].

2.2.2 Problemas de Otimização e Algoritmos

Sejam R, o conjunto dos números reais, X um conjunto qualquer no espaço e S um subconjunto de X. Define-se uma função $Z : S \to R$ tal que $x \mapsto Z(x)$ seja mínima, ou seja,

$$\pi : min\{Z(x) \mid x \in S\}$$

π é chamado problema de otimização, onde Z é a função objetivo, $x \in S$ é uma solução viável que satisfaz as restrições do problema. Um $x^* \in S$ é uma solução ótima de π se $Z(x^*) \leq Z(x)$, $\forall x \in S$. O problema também pode ser definido como um problema de maximização, ou seja, $\pi : max\{Z(x) \mid x \in S\}$ e sua solução ótima seria $Z(x^*) \geq Z(x)$, $\forall x \in S$.

Caso o conjunto $S \subset X$ seja discreto, então π é denominado problema de otimização combinatorial. Uma estratégia trivial para obtenção de soluções ótimas x^* para este problema consiste na avaliação de todas as soluções viáveis e dentro dessas soluções, a escolha da solução mínima. O inconveniente desta estratégia é que dependendo do tamanho do problema e do algoritmo utilizado, o número de combinações distintas seria extremamente grande e inviável a ser considerado, ocorrendo o que se chama de explosão combinatorial.

A teoria de complexidade fornece critérios para avaliar o grau de dificuldade de resolução de um problema de otimização combinatorial e define-se funções de complexidade tempo associados aos algoritmos utilizados. Nesta teoria, o problema é definido como uma questão geral para a qual deve ser dada uma resposta, podendo tal questão ter muitos parâmetros, cujos valores estão em aberto. Uma instância de um problema é obtida através da fixação desses valores e da especificação das propriedades que a solução do problema deve possuir.

Para um problema de otimização combinatorial a função de complexidade de tempo $f : N \rightarrow N$ de um algoritmo, expressa o tempo máximo (operações elementares) necessário para resolver qualquer instância de tamanho $n \in N$, onde N é o conjunto dos números naturais.

Seja $p(x)$ um polinômio de ordem k. Se a função complexidade tempo de um algoritmo é tal que $f(n) = a_0 + a_1x + a_2x^2 + \ldots + a_kx^k \leq p(n)$, $\forall n \in N$, este algoritmo é denominado de tempo polinomial da ordem $O(n^k)$. Assim um algoritmo de complexidade $O(n^2)$ possui complexidade menor que outro de complexidade $O(n^3)$.

Os problemas de otimização combinatorial quanto à facilidade ou dificuldade de solução classificam-se em polinomiais (P) e não-polinomiais (NP), respectivamente. Os problemas polinomiais (P) pertencem a classe de problemas cuja função complexidade tempo do algoritmo de solução exata é limitada superiormente por um polinômio em $n \in N$. Os problemas não polinomiais (NP) são aqueles para os quais não se conhecem algoritmos exatos, sendo que sua função complexidade tempo não é limitada superiormente por um polinômio em $n \in N$. Os exemplos mais conhecidos de problemas da classe (NP) são da ordem $O(n^{\log(n)})$, $O(n!)$, $O(k^n)$, etc.

Existe uma classe de problemas chamados problemas de decisão. Tais problemas

possuem apenas duas soluções possíveis, ou seja, sim ou não. Dados dois problemas de decisão, π e π' uma transformação polinomial é um algoritmo que, dado uma instância de π, produz em um tempo polinomial uma instância de π', tal que a solução de cada instância $\sigma \in \pi$ é a mesma que para a correspondente instância em π'.

Um problema de otimização não é um problema de decisão, mas pode ser transformado em um. Dado o problema de otimização π, esse problema pode ser substituído pelo problema de decisão: $\{x \in S \mid Z(x) \leq L?\}$, onde $L?$ é o conjunto de soluções para o problema de decisão.

Supondo que exista um algoritmo capaz de resolver o problema de minimização em tempo polinomial, então o problema de decisão também será resolvido em tempo polinomial, da seguinte forma: resolve-se o problema de minimização, em seguida o de decisão, comparando a solução gerada pelo primeiro com o valor de $L?$. Por outro lado, se existe um algoritmo capaz de resolver o problema de decisão em tempo polinomial, o problema de minimização também é resolvido através de sucessivos questionamentos feitos para diferentes valores de $L?$.

Um problema de decisão π é dito NP-completo, se π pertence a NP e todo problema em NP pode ser transformado em tempo polinomial para π. Como consequência dessa definição, tem-se que, se um problema NP-completo puder ser resolvido em tempo polinomial então todos os problemas NP também poderão ser. Neste sentido, os problemas NP-completos, são os problemas mais difíceis da classe NP. O interesse pelo estudo desses problemas é que a existência de problemas NP-completos sugere um meio de padronizar e resolver os problemas NP. Desta forma, o problema de programação inteira é o problema NP-completo mais utilizado como padrão de resolução de problemas em NP. Vários problemas combinatoriais, como o problema de cobertura, problema de partição, problema da mochila, quando transformados para problemas de programação inteira apresentam boas condições de resolução.

Um problema π é chamado NP-fácil se existe um problema $\pi' \in$ NP tal que π pode ser reduzido a π'. Um problema π é chamado NP-difícil se existe um problema de decisão π' NP-completo tal que π' pode ser reduzido a π.

Uma complementação sobre otimização combinatorial e complexidade pode ser vista em Buriol (2000)[7], Garey e Johnson (1979)[17] e Grotschel et al. (1988)[19].

2.3 Heurísticas

Como técnicas de otimização têm-se os métodos exatos que garantem a solução ótima, porém possuem uma modelagem mais complexa podendo gastar um tempo proibitivo para gerar tal solução e nem sempre conseguem produzir uma solução viável rapidamente, (Cormen et al., 2002) [12]. Problemas de Otimização Combina-

tória possuem universo grande de dados, pois existe um número muito extenso de combinações a serem analisadas o que torna inviável a aplicação de métodos exatos. Neste contexto, outra técnica bastante utilizada é conhecida como *algoritmos heurísticos*, que ganham importância e hoje representam uma gama de opções muito viáveis para solução de problemas de Otimização Combinatória, (Silva, 2005) [44].

Heurística, significa descobrir, é o termo utilizado para descrever um método que, baseado na experiência ou julgamento, parece conduzir a uma boa solução de um problema, mas que não garante produzir uma solução ótima (Foulds, 1984) [16].

Como foi visto anteriormente, para problemas de Otimização Combinatória os métodos exatos garantem a determinação da solução ótima, mas são inviáveis de serem implementados devido a sua complexidade. Nesse caso, aplicam-se métodos de aproximação, chamados métodos heurísticos, que são procedimentos simples, na maioria das vezes intuitivos, e que produzem soluções consideradas satisfatórias em um bom tempo computacional. Uma solução ótima de um problema nem sempre é o alvo dos métodos heurísticos, uma vez que, tendo como ponto de partida uma solução viável, baseiam-se em sucessivas aproximações direcionadas a um ponto ótimo.

A importância dos métodos heurísticos é originada do bom desempenho médio - detectado de forma predominantemente experimental - desses métodos, quando aplicados ao tratamento de problemas (normalmente NP-difíceis) para os quais não são conhecidos métodos eficientes que forneçam garantias (Sucupuira, 2012)[46].

O sucesso de uma heurística depende de sua capacidade de: adaptação a instâncias especiais; escapar de ótimos locais; fazer uso da estrutura do problema; estrutura eficiente de dados; pré-processamento; boas técnicas para construir soluções iniciais; reinicializar procedimentos; melhoria de solução através de busca local; randomização controlada; diversificação de busca quando nenhuma melhoria adicional parece possível; intensificação da busca em regiões promissoras, (Rodrigues, 2012) [40].

Frequentemente na literatura encontra-se o termo meta-heurística, que na computação possui diferentes definições, como, por exemplo: “Uma meta-heurística é um conjunto de conceitos que podem ser usados para definir métodos heurísticos, os quais podem ser aplicados a um conjunto amplo de diferentes problemas. Em outras palavras, uma meta-heurística pode ser vista como uma ferramenta algorítmica geral que pode ser aplicada a diferentes problemas de otimização, com modificações relativamente pequenas para torná-la adaptável a um problema específico” [30].

De acordo com Becceneri (2008)[3], a grande importância na aplicabilidade de uma meta-heurística é o balanço dinâmico entre diversificação e intensificação, fazendo uma distinção entre os termos ingleses exploration e exploitation. O primeiro pode-se traduzir por diversificação, exploração diversificada, busca em largura ou simplesmente exploração; o segundo por exploração focada, busca em profundidade ou intensificação. Um dos desafios na aplicação de uma meta-heurística é encontrar

o equilíbrio ideal entre diversificação e intensificação.

Baseada na simples ideia de que uma solução ótima local pode ser melhorada é que são aplicados procedimentos de busca local em uma solução que pode ser obtida fazendo a exploração no espaço de soluções. O *Simulated Annealing*, um método de busca não populacional, explora o espaço de soluções por meio de movimentos, os quais são aplicados, a cada passo, sobre a solução corrente, gerando outra solução promissora em sua vizinhança. Já o Algoritmo Genético, um método baseado em busca populacional, consiste em manter um conjunto de boas soluções e procura combiná-las, de forma a tentar produzir soluções ainda melhores.

O objetivo dessas duas heurísticas é tentar melhorar a solução, para isso utiliza-se o conceito de estrutura de vizinhança para explorar o espaço de soluções do problema de otimização na busca de melhores soluções.

A principal estratégia de busca dos dos dois métodos é tentar escapar de pontos ótimos locais, a fim de encontrar um possível ponto ótimo global. A Figura 2.2 mostra a exploração do espaço de busca apresentando dois pontos mínimos locais e um ponto ótimo global.

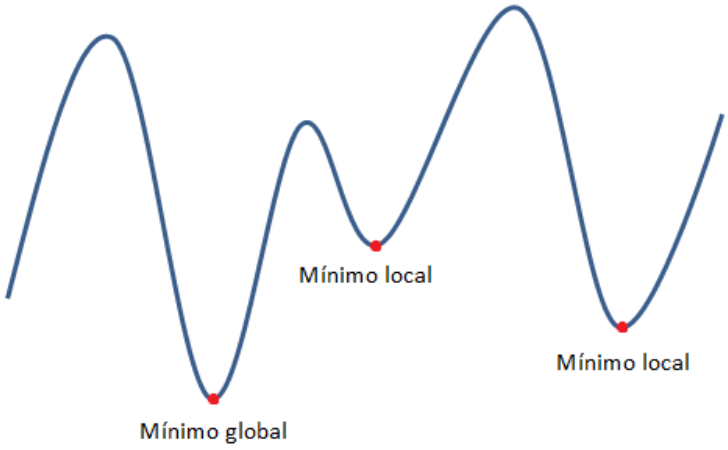


Figura 2.2: *Espaço de busca - adaptado de Becceneri [3].*

Para efeito deste trabalho, os métodos de solução empregados serão tratados como heurísticas acompanhando assim os autores em que o estudo se baseia.

As considerações realizadas neste capítulo são necessárias para o entendimento do problema de alocação de berços e sua modelagem apresentados no capítulo a seguir.

Capítulo 3

Descrição do Problema

Neste capítulo será apresentado uma breve descrição do transporte marítimo e sua importância econômica, contextualizando, assim, o cenário onde ocorre um dos principais problemas encontrados no sistema portuário, o problema de alocação em berços.

3.1 Transporte Marítimo

Dentre os meios de transportes, o mais antigo é o transporte marítimo, pois é utilizado desde a antiguidade. No entanto, seu incremento aconteceu efetivamente após o término da Primeira Guerra Mundial, resultado de grandes inovações no campo tecnológico, as quais levaram a significativas evoluções. Dentre as mudanças, se pode destacar a melhora em relação à capacidade de carga a ser transportada nos navios, além da criação de embarcações específicas, especializadas no transporte de um determinado tipo de carga.

O processo de modernização portuária, implementado no Brasil a partir da Lei de Modernização dos Portos (Lei 8.630/93) [6], buscou solucionar os problemas gerados pelos altos custos, baixa produtividade, serviços não competitivos, excesso de pessoal, subsídios e burocracia governamental. Todas essas características configuram o "modelo portuário latino-americano tradicional", e levam a um intenso processo de reestruturação e de reformas nos portos, objetivando a compatibilidade com o acelerado crescimento do comércio entre os países e os blocos econômicos e a demanda por uma produção mais eficiente, (Kitzmann, 2010) [23].

Faz-se necessário, então, que haja uma modernização das estruturas físicas do porto, bem como de processos gerenciais dando suporte às operações portuárias. A operação portuária pode ser definida como o conjunto de todas as operações para realizar a passagem da mercadoria, desde o transporte marítimo até o transporte terrestre e vice-versa. O objetivo da operação portuária é sempre buscar a maior eficiência e eficácia. Em outras palavras, isso quer dizer minimizar os custos de

transporte e armazenagem, e aumentar o fluxo (movimentação de cargas) dado um determinado período, (Oliveira, 2011) [36].

A competitividade de um terminal portuário está intimamente ligada ao planejamento e a logística de operações. De acordo com Botter et al. [5], os níveis de planejamento e aspectos de tomada de decisão relacionados ao subsistema ou área de operações de navios, na divisão de planejamento de atracação e desatracação podem ser divididos em:

Estratégico: definição da quantidade de berços a serem disponibilizados e tamanho de cais, profundidade e calado de projeto dos berços, forma de construção e projetos de ampliação de cais.

Tático: o acompanhamento do crescimento do calado de navios que operam nos tráfegos atendidos pelo terminal, manutenção da profundidade do berço por meio de acompanhamento dos programas de dragagem alinhados com os dados fornecidos pelos armadores e informações técnicas de projeto.

Operacional: a alocação dos navios aos berços de acordo com as regras de atracação definidas entre o terminal e o armador. Essas regras de atendimento são normalmente FCFS (*First Come, First Served*), contudo o uso de janelas de tempo de atracação é cada vez mais solicitado pelos clientes e fornecido pelos terminais, o que permite melhor organização da distribuição de berços, equipamentos e ternos de trabalho. A área de armazenamento da carga do navio em questão e da ocupação linear de cais dos berços pelos navios em operação deve ser considerada.

O cenário brasileiro no contexto portuário, no ano de 2010, caracterizou-se pelo retorno do crescimento da movimentação de cargas. A maioria das instalações portuárias incrementou a sua movimentação de carga em relação ao ano anterior. Tal desempenho pode ser explicado pela conjuntura favorável à retomada do crescimento econômico em vários setores, o que impulsionou o aumento da demanda e gerou reflexos na movimentação de cargas, (ANTAQ) [1]. Dessa forma, cresce no meio acadêmico o estudo de problemas que envolvem a operação portuária, no intuito de contribuir para o bom desenvolvimento e crescimento da movimentação de cargas.

Dentre os problemas existentes no sistema portuário, destacam-se:

- Problema de aquisição e/ou locação de equipamentos;
- Problema de alocação de guindastes;
- Problema de dimensionamento de berços;
- Problema de *layout* do porto;

- Problema de alocação de navios em berços.

3.2 O Problema de Alocação em Berços

3.2.1 Considerações Gerais

O Problema de Alocação em Berços - PAB está acondicionado no nível de planejamento da operação portuária e deve obedecer a determinadas regras e acordos entre o terminal (entidade prestadora de serviços de operação nos portos) e o armador (entidade que procede ao armamento do navio). Devido a aleatoriedade nas chegadas dos navios, por razões contratuais, as regras de atracação obedecem em geral o sistema FCFS (*First Come, First Served*), respeitando a alocação de navios por metro linear de cais, ou seja, a disponibilidade de berço de atracação. Essas atracações, seja em cais público ou privado, na maioria das vezes é feita por ordem de chegada do navio, por prioridade e condição de atracação, isto é, de acordo com a janela de tempo oferecida pelo terminal ao armador, considerando um sistema total de custos dos navios em fila.

A janela de tempo de atracação é um período de tempo em horas oferecido pelo terminal ao armador em um determinado dia para que este atraque o seu navio com a garantia de reserva de berço para atracação. Isso garante o pagamento de penalidades pelo terminal caso o navio chegue à janela determinada e não possa atracar em virtude de não haver berço ou espaço de cais disponível.

As situações analisadas nas regras de atracação de chegada dos navios são as seguintes:

O navio chega antes da janela de tempo acordada: o armador arcará com todos os custos de espera até o início de sua janela de tempo, e o terminal poderá ou não atendê-lo antes do início da sua janela de tempo, dependendo da disponibilidade e da programação de seus berços.

O navio chega dentro (durante) a janela de tempo: o terminal deve conceder a atracação imediata do navio em um berço que atenda às suas características de comprimento e calado. Se não for possível a atracação do navio em virtude de problemas operacionais com os navios anteriores (exemplos: atraso nas programações dos navios atracados, falta de carga no costado, quebra de equipamento de operação de cais ou retaguarda), todos os custos da espera do navio da vez são de responsabilidade do terminal, independente do ocorrido.

O navio chega após a janela de tempo estipulada: o terminal está livre das penalidades de espera e atenderá esse navio quando houver espaço na programação dos berços ou for possível violar ou relaxar a fila, isto é, atracar o navio desde que atenda

às condições de comprimento de berço e cuja somatória de tempos seja menor ou igual à do navio atracado, (Botter et al.) [5].

3.2.2 Descrição do PAB

O Problema de Alocação em Berços *(Berth Allocation Problem)* consiste em alocar os navios que chegam a um determinado porto nas posições de atracações disponíveis ao longo de um cais (berços). Esse problema enfrenta duas decisões inter-relacionadas: onde e quando os navios devem atracar. Os navios que chegam ao porto irão atracar no berço mais conveniente, ou em um berço livre que possa recebê-los. Caso não haja berços livres adequados à operação do navio em questão, este navio irá para uma fila de navios aguardando atracação. Deste modo, o tempo que o navio fica aguardando um berço de atracação em fila é o parâmetro que se utiliza como principal nível de serviço na gestão portuária (Cordeau, 2005)[11].

A Figura 3.1 apresenta um cenário para o PAB com quatro berços ocupados, um navio aguardando atendimento e um outro navio chegando ao porto.

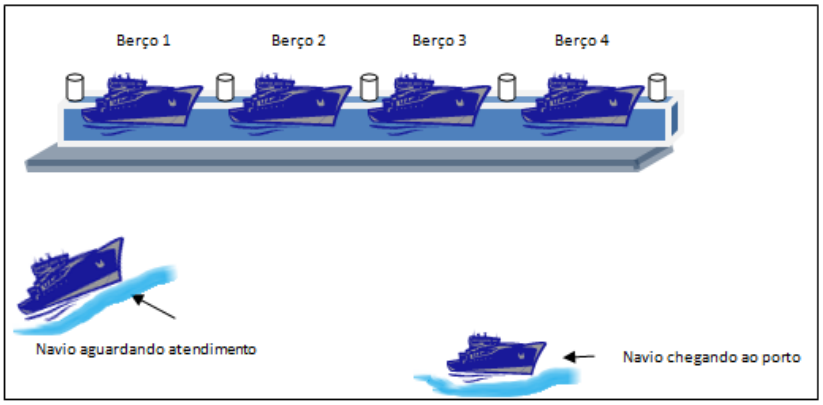


Figura 3.1: *Cenário usual para o PAB - adaptado de Mauri [28].*

O PAB possui então, como principal objetivo minimizar os custos referentes ao porto e ao armador, que é relacionado ao tempo de serviço, nesse caso o tempo total de atendimento, considerado desde o momento da chegada do navio ao porto, sua atracação e saída do berço.

Em relação ao local de atracação, na dimensão espacial, há restrições em relação à profundidade da água (calado do berço) e com a distância máxima em relação à localização mais favorável ao longo do cais, calculadas com relação à localização da saída do contêiner e para o espaço reservado para a entrada do contêiner. Na dimensão temporal, as restrições são expressas como "janelas de tempo" para o tempo total de conclusão de serviço do navio. Algumas janelas de tempo são suaves e podem ser relaxadas, com um custo adequado, (Cordeau, 2005) [11].

Outra decisão que antecede o PAB é conhecida como "Problema de Atribuição

de Guindastes - PAG" e afeta diretamente o tempo de atendimento dos navios. O PAG diz respeito aos tipos de máquinas utilizadas para efetuar a carga e a descarga, ou mesmo o número de guindastes disponíveis no berço (Mauri, 2008)[28].

A importância do PAB é aparente com o aumento da frequência de transporte entre terminais portuários. Se um número suficiente de berços não pode ser garantido para a entrada dos navios, o tempo desses navios atrasados crescerá, o que por sua vez, afeta a eficiência e, consequentemente, a produtividade do porto. Isso também afeta a competitividade: se a entrada dos navios necessita de uma longa espera, o nível de serviço oferecido ao armador não será suficiente. Portanto, minimizar o tempo de espera de entrada dos navios é um elemento crucial do PAB. Além disso, o tempo de atendimento de cada navio influencia na eficiência do porto. Se o tempo de tratamento é reduzido, a produtividade no terminal aumenta reforçando o efetivo gerenciamento de custos do terminal portuário, (Rodrigues, 2012) [40].

3.3 Modelagem do PAB

O modelo matemático para o PAB que motivou diversos estudos, foi observado por Legato et. al. (2001) [33] que o modelou como um Problema de Roteamento de Veículos com Garagens Múltiplas e Janelas de Tempo. No entanto, Mauri (2008) [28] reformula o modelo como um Problema de Roteamento de Veículos com Garagens Múltiplas SEM Janelas de Tempo, cuja resolução é menos árdua com relação ao modelo descrito por Cordeau. Primeiramente apresenta-se o problema clássico de roteamento de veículos (PRV) e o problema de roteamento de veículos com múltiplas garagens (PRVMG).

Formalmente, o problema (PRV) é definido com base em um grafo $G = (V; A)$, onde $V = \{v_1, v_2, \ldots, v_m\} \cup V_0$ é um conjunto de vértices e $A = \{e_1, e_2, \ldots, e_n\}$ é um conjunto de arestas. Cada vértice $v_i \in V - V_0$ representa um cliente a ser atendido, sendo que $v_0 \in V_0$ representa a garagem. Por sua vez, cada aresta $(i, j) \in A$ está associada a um custo não-negativo c_{ij}, normalmente a distância entre dois vértices. É importante ressaltar que todas as rotas têm garagem (v_0) como ponto de partida e chegada e inclui um subconjunto de arestas de A. Cada cliente tem uma demanda $q \geq 0$ a ser atendida por algum dos r veículos inicialmente estacionados na garagem e só pode ser visitada apenas uma vez. O PRV consiste em determinar um conjunto de rotas de modo a minimizar a soma dos custos atribuídos às arestas de A. Além disso, ressaltamos que para cada rota a capacidade Q do veículo associado deve ser respeitada. O problema de roteamento de veículos com múltiplas garagens (PRVMG) é uma generalização do PRV, onde o conjunto de vértices V pode ser definido por $V = \{v_1, v_2, \ldots, v_m\} \cup V_0$ onde $V_0 = \{v_{01}, v_{02}, \ldots, v_{0g}\}$ são as garagens. Uma rota i pode ser definida por $R_i = \{g, v_1, v_2, \ldots, v_n, g\}$ com $g \in V_0$ e $n \leq m$. O custo de

uma rota pode ser calculado como o PRV clássico.

Assim, no modelo PAB, os navios são tratados como clientes e os berços como garagens ou depósitos (cada um com seu veículo específico). Existem m veículos (uma para cada garagem), sendo que cada um inicia e termina sua rota na sua própria garagem. Os navios são modelados como vértices em um multigrafo, onde cada garagem (berço) ainda é dividida em um vértice de origem e um de destino. Esses vértices de origem e destino são criados no modelo a fim de fazer corresponder ao período de funcionamento dos berços.

O modelo descrito por Cordeau (2001) [10] é tratado em sua forma discreta e pode ser representado por um multigrafo $G^k = (V^k, A^k)$, $\forall k \in M$, onde $V^k = N \cup o(k), d(k)$ e $A^k \subseteq V^k \times V^k$, sendo M o conjunto de berços e N o conjunto de navios.

Cada berço k, poderá ser representado por um super vértice que possui dois vértices elementares $o(k)$ e $d(k)$ correspondente ao período de funcionamento do berço k. A rota R para o berço k é o conjunto $R_k = \{o(k), v_1, v_2, \ldots, v_n, d(k)\}$, onde o navio v_1 é o primeiro navio a ser atendido no berço k enquanto que o navio v_n é o último navio a ser atendido no berço k. No problema de alocação em berços os seguintes conjuntos de variáveis e constantes são considerados:

N conjunto de navios;

M conjunto de berços;

$x_{ij}^k \in \{0,1\} \forall k \in M, \forall (i,j) \in A^k$, $x_{ij}^k = 1$ se o navio j é atendido pelo berço k após o navio i;

$T_i^k \forall k \in M, i \in N$, horário que o navio i atracou no berço k;

$T_{o(k)}^k \forall k \in M$, horário que o primeiro navio atracou no berço k;

$T_{d(k)}^k \forall k \in M$, horário que o último navio desatracou no berço k;

t_i^k, duração do atendimento do navio i no berço k;

a_i, horário de chegada para o navio i;

b_i, horário de término de janela de tempo para o navio i;

v_i, valor (custo) de tempo de serviço para o navio i;

s^k, horário de abertura do berço k;

e_k, horário de fechamento do berço k;

$M_{ij} = max\{b_i + t_i^k - a_j, 0\}$, $\forall k \in M$ e $\forall (i,j) \in N$.

Onde a modelagem segundo Cordeau (2001) [10] é:

Minimizar:

$$Z = \sum_{i \in N} \sum_{k \in M} v_i \left[T_i^k - a_i + t_i^k \sum_{j \in N \cup \{d(k)\}} x_{ij}^k \right] \tag{3.1}$$

Sujeito à:

$$\sum_{k \in M} \sum_{j \in N \cup \{d(k)\}} x_{ij}^k = 1, \qquad \forall i \in N \tag{3.2}$$

$$\sum_{j \in N \cup \{d(k)\}} x_{o(k),j}^k = 1 \qquad \forall k \in N \tag{3.3}$$

$$\sum_{i \in N \cup \{o(k)\}} x_{i,d(k)}^k = 1 \qquad \forall k \in N \tag{3.4}$$

$$\sum_{j \in N \cup \{d(k)\}} x_{i,j}^k - \sum_{j \in N \cup \{o(k)\}} x_{j,i}^k = 0 \qquad \forall k \in M, \forall i \in N \tag{3.5}$$

$$T_i^k + t_i^k - T_j^k \leq (1 - x_{ij}^k) M_{ij}^k \qquad \forall k \in M, \forall (i,j) \in A^k \tag{3.6}$$

$$T_i^k \geq a_i \qquad \forall k \in M, \forall i \in N \tag{3.7}$$

$$T_i^k + t_i^k \sum_{j \in N \cup \{d(k)\}} x_{ji}^k - b_i \qquad \forall k \in M, \forall i \in N \tag{3.8}$$

$$T_{o(k)}^k \geq s^k \qquad \forall k \in M \tag{3.9}$$

$$T_{d(k)}^k \leq e^k \qquad \forall k \in M \tag{3.10}$$

$$x_{ij}^k \in \{0,1\} \qquad \forall k \in M, \forall (i,j) \in A^k \tag{3.11}$$

Onde:

A equação (3.1) representa a função objetivo que consiste em minimizar o tempo decorrido desde o momento em que os navios chegam, atracam e são atendidos, correspondendo ao custo total da alocação. A restrição (3.2) garante que cada navio é alocado somente a um único berço, onde o navio de índice j é atendido após o navio de índice i no berço k. A restrição (3.3) garante que somente um navio será o primeiro a ser atendido em cada berço, ou seja, existe somente um navio de índice j que estará unido ao vértice de origem $o(k)$. Analogamente a restrição (3.4) garante que um único navio também será o último a ser atendido em cada berço, esse navio de índice i estará unido ao vértice de destino $d(k)$. A restrição (3.5) garante a conservação de fluxo para os demais navios, ou seja, o atendimento para os navios alocados ao berço k, que deverão entrar e sair do respectivo berço. A restrição (3.6)

determina o cálculo do horário de atracação dos navios que podem ser atendidos nos berços, ou seja, o berço deverá ser capaz de atender os navios. As restrições (3.7) e (3.8) garantem que o horário de atracação seja após a chegada do navio, e que o horário do término de atendimento do navio seja anterior ao horário limite do navio (janela de tempo). As restrições (3.9) e (3.10) garantem a não violação das janelas de tempo nos berços. Finalmente a restrição (3.11) garante que as variáveis de decisão sejam binárias.

A Figura 3.2, mostra para cada navio, os tempos descritos no modelo proposto por Cordeau (2001) [10].

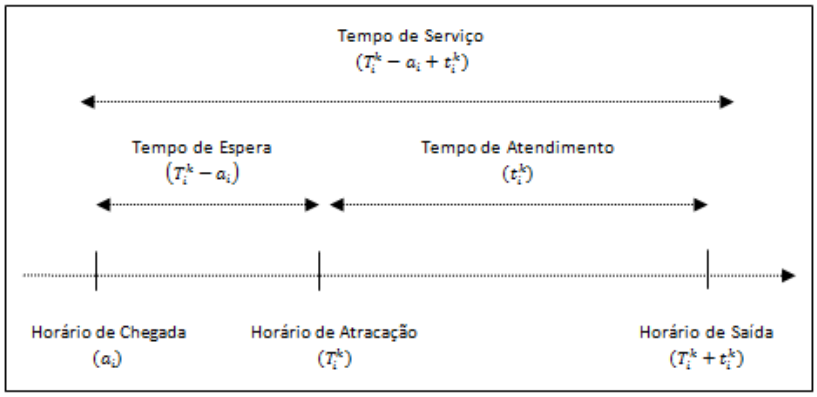


Figura 3.2: *Variáveis referentes ao tempo.*

3.3.1 Reformulação do modelo

Neste trabalho utiliza-se a reformulação proposta por Mauri (2008) [28], uma vez que as heurísticas aqui utilizadas são ferramentas algorítmicas gerais, que com modificações relativamente pequenas tornam-se adaptáveis a diferentes problemas de otimização. No modelo reformulado por Mauri (2008) [28], as restrições 3.7 e 3.8 são relaxadas, sendo transferidas para a função objetivo 3.13. Analogamente, as restrições 3.9 e 3.10 também são transferidas para a função objetivo 3.14. As demais restrições são mantidas, porém, na função objetivo são adicionados fatores de penalização $w = [w_0, w_1, w_2]$ para cada expressão. O novo modelo proposto é apresentado a seguir:

Minimizar:

$$Z^* = w_0 \sum_{i \in N} \sum_{k \in M} v_i \left(T_i^k - a_i + t_i^k \sum_{j \in N \cup \{d(k)\}} x_{ij}^k \right) + \tag{3.12}$$

$$w_1 \sum_{i \in N} \sum_{k \in M} \sum_{j \in N \cup \{d(k)\}} x_{ij}^k \left(max(0, a_i - T_i^k) + max(0, T_i^k + t_i^k - b_i) \right) + \tag{3.13}$$

$$w_2 \sum_{k \in M} \left(max(0, s^k - T^k_{o(k)}) + max(0, T^k_{d(k)} + e^k) \right) \tag{3.14}$$

Sujeito à:

$$\sum_{k \in M} \sum_{j \in N \cup \{d(k)\}} x^k_{ij} = 1, \qquad \forall i \in N \tag{3.15}$$

$$\sum_{j \in N \cup \{d(k)\}} x^k_{o(k),j} = 1 \qquad \forall k \in N \tag{3.16}$$

$$\sum_{i \in N \cup \{o(k)\}} x^k_{i,d(k)} = 1 \qquad \forall k \in N \tag{3.17}$$

$$\sum_{j \in N \cup \{d(k)\}} x^k_{i,j} - \sum_{j \in N \cup \{o(k)\}} x^k_{j,i} = 0 \qquad \forall k \in M, \forall i \in N \tag{3.18}$$

$$T^k_i + t^k_i - T^k_j \leq (1 - x^k_{ij}) M^k_{ij} \qquad \forall k \in M, \forall (i,j) \in A^k \tag{3.19}$$

$$x^k_{ij} \in \{0, 1\} \qquad \forall k \in M, \forall (i,j) \in A^k \tag{3.20}$$

Nesse modelo, pode-se notar que o tempo de serviço (com seu valor de custo associado) é representado na expressão (3.12). A expressão (3.13) minimiza as violações nas janelas de tempo dos navios. Já a expressão (3.14) minimiza as violações nas janelas de tempo dos berços.

Analisando as restrições do modelo acima, trata-se de um Problema de Roteamento de Veículos com Garagens Múltiplas SEM Janelas de Tempo, ou seja, um problema cuja resolução é menos árdua em relação ao modelo descrito anteriormente (com janelas de tempo). O modelo (3.12 a 3.20) pode resultar em soluções inviáveis para o PAB, porém essas inviabilidades são eliminadas através da penalização imposta.

A solução inicial para o problema é gerada através de uma heurística de distribuição de berços e programação de navios. A heurística de distribuição de berços aos navios é aleatória, porém existe a necessidade de verificar se o berço selecionado poderá atender o navio em questão. Na heurística de programação são efetuados os cálculos de horário de atracação de cada navio e da função objetivo da solução das equações (3.12), (3.13) e (3.14).

Os resultados obtidos por Mauri através do modelo apresentado são encorajadores, motivam o estudo sobre algoritmos de resolução para o PAB, bem como suas possíveis aplicações a problemas reais encontrados em portos brasileiros.

Capítulo 4

Simulated Annealing e Algoritmo Genético aplicados ao PAB

Introdução

Os Algoritmos Genético (AG) e o *Simulated Annealing* (SA) são algoritmos projetados para encontrar o máximo ou mínimo de uma função que representa alguma característica do processo que está sendo modelado. Esses algoritmos possuem mecanismos que os fazem escapar de pontos de ótimos locais, entretanto, a evolução desses algoritmos no tempo se dá de forma completamente diferente. O SA no seu processo de busca trabalha com apenas um ponto, gerando a partir deste sempre uma nova solução que é testada e que pode ser aceita ou não. Já o AG trabalha com um conjunto de pontos, chamado de população, do qual gera outra população que sempre é aceita. Em comum nesses dois algoritmos, tem-se a forma como o próximo ponto ou a próxima população é gerada obedecendo propriedades estocásticas, (nETO, 2010) [34].

Segundo Kirkpatrick (1983) [22], existem duas estratégias básicas para heurísticas: "dividir e conquistar" e melhoramentos iterativos. No primeiro, divide-se o problema em subproblemas de tamanho administrável e então resolvem-se os subproblemas. As soluções dos subproblemas devem ser unidas de alguma forma para obter a solução do problema original. Para este método produzir soluções muito boas, os subproblemas devem ser naturalmente disjuntos, e a divisão deve ser de forma apropriada a fim de que os erros cometidos na junção das soluções não prejudiquem os ganhos obtidos na aplicação de métodos mais poderosos a solução dos subproblemas. No melhoramento iterativo se começa com o sistema em uma configuração conhecida. Uma operação de reorganização padrão é aplicada em todas as partes do sistema até que uma configuração reorganizada, que melhore a função custo, seja descoberta. A configuração reorganizada então se torna a nova configuração do

sistema, e o processo continua até que nenhuma outra melhora seja encontrada. O melhoramento iterativo consiste em uma busca no espaço coordenado através de passos de reorganização das configurações, que leve para valores menores da função. Visto que esta busca usualmente fica presa em um mínimo local que não é um mínimo global, é costume realizar várias vezes o processo, começando em diferentes configurações aleatoriamente geradas, e salvando o melhor resultado. Como uma outra alternativa, surgiram algoritmos que possuem mecanismos para impedir que essa busca fique presa em ótimos locais, dentre eles estão, o Algoritmo Genético e o Algoritmo *Simulated Annealing*, os quais foram criados como heurísticas de processo naturais.

Estes algoritmos trabalham bem na prática e apresentam vantagens como versatilidade, eficiência e simplicidade. É crescente o interesse nesta classe de algoritmos pela sua facilidade de implementação e, principalmente, pela sua robustez para resolver uma grande variedade de problemas. São algoritmos iterativos de propósito geral largamente utilizados para problemas de otimização, em especial, otimização combinatória, [42].

Muitas pesquisas tem sido desenvolvidas em testes competitivos de novas implementações desses algoritmos, ou na investigação do porquê do fracasso ou sucesso destes algoritmos frente a alguns problemas. Nesse trabalho, esses algoritmos foram estudados para resolver um problema importante de otimização combinatória conhecido na literatura como o Problema de Alocação em Berços.

4.1 Simulated Annealing

A heurística *Simulated Annealing* (SA) é uma estratégia de busca, podendo ser aplicada a diferentes problemas de otimização, que incorpora mecanismos que possibilitam sair de pontos ótimos locais, permitindo a busca de soluções em regiões mais promissoras. O grande desafio e importância na sua aplicabilidade consiste em encontrar o equilíbrio entre a diversificação (exploração diversificada ou busca em largura) e intensificação (exploração focada ou busca em profundidade), (Becceneri, 2008) [3].

O SA surgiu do algoritmo denominado METROPOLIS (1953)[31] e Kirkpatrick et al. (1983) [22] sugeriram sua utilização no ambiente computacional. O SA simula o resfriamento de um conjunto de átomos aquecidos (recozimento). Essa simulação corresponde a um processo térmico de liquidificação de um cristal em alta temperatura. A temperatura sofre uma lenta e gradativa diminuição até um estado de energia ou ponto de solidificação mínima. O SA é considerado um dos primeiros algoritmos a encontrar alternativas para escapar do mínimo local. A ideia básica é permitir que soluções piores que a atual sejam selecionadas para exploração da vizinhança,

escapando assim, do mínimo local. A probabilidade de ser selecionado um estado pior vai diminuindo ao longo da busca devido ao fato da redução de temperatura. O algoritmo possui uma estrutura padrão que pode ser modificada, além dos parâmetros iniciais que devem ser calibrados antes da execução. Essas peculiaridades geram diversas maneiras de implementação e execução do algoritmo.

De acordo com Chaves [9], a cada iteração do método, um novo estado é gerado a partir do estado corrente por uma modificação aleatória neste; se o novo estado é de energia menor que o estado corrente, esse novo estado passa a ser o estado corrente; se o novo estado tem uma energia maior que o estado corrente em Δ unidades, a probabilidade de se mudar do estado corrente para o novo estado é: $e^{\frac{-\Delta}{kT}}$, onde k é a constante de Boltzmann e T é a temperatura corrente. Este procedimento é repetido até se atingir o equilíbrio térmico.

No início do processo, a temperatura é elevada e a probabilidade de se aceitar soluções de piora é maior, as soluções de piora são aceitas para escapar de ótimos locais. A probabilidade de se aceitar uma solução de piora depende de um parâmetro, chamado temperatura, quanto menor a temperatura, menor a probabilidade de se aceitar soluções de piora. Atingido o equilíbrio térmico, a temperatura é diminuída e assim a taxa de aceitação de movimentos de piora é, portanto, diminuída com o decorrer das iterações.

No final do processo, praticamente não se aceita movimentos de piora e o método se comporta como o método da descida/subida, o final do processo se dá quando a temperatura se aproxima de zero e nenhuma solução de piora é mais aceita, evidenciando o encontro de um ótimo local.

Segundo Blum e Roli (2001) [4] outro fator importante nessa heurística é o processo de resfriamento. Programações de resfriamento que garantem a convergência para um ótimo global infelizmente são impraticáveis, pois necessitam de um tempo infinito para se atingir o ótimo global. Entretanto, programações de resfriamento mais rápidas são adotadas. Uma das mais usadas segue uma lei geométrica: $T_{k+1} = \alpha T_k$, no qual k se refere à iteração e $\alpha \in (0, 1)$, que corresponde a um decréscimo exponencial da temperatura. Geralmente o valor de α oscila entre $0,90$ e $0,99$.

A regra de resfriamento pode variar durante a busca, com o objetivo de ajustar o balanço entre diversificação e intensificação. Por exemplo, no início da busca, a temperatura T pode ser constante ou decrescer linearmente, favorecendo assim mais a diversificação. Então, T pode seguir uma regra, como a geométrica, para convergir para um mínimo local no fim da busca, ou seja, mais intensificado do que diversificado.

Escolher os valores iniciais dos parâmetros é fundamental para o sucesso do algoritmo. Os parâmetros são formados pela temperatura inicial T_0, e a temperatura

final ou ponto de parada T_c, número de iterações para selecionar o novo estado a cada valor de temperatura SA_{max}, esse número de iteração também é chamado de Ponto de Equilíbrio, e ainda o índice α de resfriamento de temperatura. Esses valores bem calibrados aumentam a chance de localizar um estado ótimo global.

A seguir é apresentado o algoritmo clássico *Simulated Annealing* (Figura 4.1).

1. Atribui-se à S uma solução inicial
2. $S^* \leftarrow S$; (Melhor solução obtida até então)
3. $IterT \leftarrow 0$; (Número de iterações na temperatura T)
4. $T \leftarrow T_0$; (Temperatura corrente)
5. ENQUANTO $(T > T_c)$ FAÇA
6. ENQUANTO $(IterT < SA_{max})$ FAÇA
7. $IterT \leftarrow IterT + 1$;
8. GERAR (um vizinho qualquer $S' \in \Psi(S)$);
9. $\Delta \leftarrow f(S') - f(S)$;
10. SE $(\Delta < 0) S \leftarrow S'$;
11. SE $(f(S') < f(S^*)) S^* \leftarrow S$; FIM SE
12. SENÃO
13. TOMAR $(x \in [0, 1])$;
14. SE $(x < e^{\frac{-\Delta}{T}}) S \leftarrow S'$; FIM SE
15. FIM SE
16. FIM ENQUANTO
17. $T \leftarrow \alpha * T$; $IterT \leftarrow 0$;
18. FIM ENQUANTO
19. $S \leftarrow S^*$;
20. RETORNAR (S).

Figura 4.1: *Algoritmo clássico - Simulated Annealing.*

O algoritmo descrito começa com uma solução inicial que poderá ser obtida de forma aleatória ou por uma estratégia que utilize algum outro método.

O parâmetro T_0, é um valor específico para cada problema, e deverá ser suficientemente grande para que todas as transições sejam inicialmente aceitas. Existem muitas propostas para o cálculo da temperatura inicial e sugestões podem ser encontradas em Becceneri (2008) [3].

O procedimento principal consiste em um *loop* que gera ou (escolhe), em cada

iteração, um único vizinho S' da solução corrente S. Essa função de gerar ou (escolher) um vizinho é essencial ao bom desempenho do algoritmo, pois se analisarmos muitos vizinhos, podemos comprometer o tempo de processamento (intensificação e diversificação) do algoritmo. A cada geração de um vizinho S' de S ($S' \in \Psi(S)$), é testada a variação do valor da função objetivo (custo), isto é $\alpha = f(S') - f(S)$. Para um problema de minimização como é o caso específico do PAB, se $\Delta < 0$ o método aceita a solução e, S' passa a ser a nova solução corrente. Caso $\Delta \geq 0$ a solução vizinha candidata também poderá ser aceita, mas neste caso, com uma probabilidade $e^{\frac{-\Delta}{T}}$, onde T é um parâmetro do método chamado de temperatura, que regula a probabilidade de aceitação de uma solução de pior custo. A temperatura T inicialmente assume um valor elevado T_0 e após um número fixo de iterações SA_{max} (o qual representa o número de iterações necessárias para o sistema atingir o equilíbrio térmico em uma dada temperatura), a temperatura é gradativamente diminuída por uma razão de resfriamento α, tal que, em um instante t, $T_t \leftarrow \alpha T_{t-1}$, sendo $0 < \alpha < 1$. Com esse procedimento, dá-se uma chance maior para escapar de mínimos locais e, à medida que T aproxima-se de zero, o algoritmo comporta-se como um método de descida, uma vez que diminui a probabilidade de se aceitar movimentos de piora ($T \to 0 \Rightarrow e^{\frac{-\Delta}{T}} \to 0$).

O procedimento termina quando a temperatura chega a um valor próximo de zero (temperatura de congelamento: T_c) e nenhuma solução que piore o valor da melhor solução é mais aceita, isto é, quando o sistema atinge a estabilidade.

Segue o algoritmo *Simulated Annealing* modificado (Figura 4.2) para o problema específico de alocação em berços adaptado por Mauri (2008) [28].

1. DADO $(\alpha, SA_{max}, T_0, T_c)$ FAÇA
2. GERAR (uma solução S através da heurística de distruibuição);
3. AVALIAR (a solução S através da heurística de programação);
4. $S^* \leftarrow S$; (Melhor solução obtida até então)
5. $IterT \leftarrow 0$; (Número de iterações na temperatura T)
6. $T \leftarrow T_0$; (Temperatura corrente)
7. ENQUANTO $(T > T_c)$ FAÇA
8. ENQUANTO $(IterT < SA_{max})$ FAÇA
9. $IterT \leftarrow IterT + 1$;
10. GERAR (um vizinho qualquer S' através dos movimentos de troca);
11. AVALIAR (a solução S' através da heurística de programação);
12. $\Delta \leftarrow f(S') - f(S)$;
13. SE $(\Delta < 0) S \leftarrow S'$;
14. SE $(f(S') < f(S^*)) S^* \leftarrow S$; FIM SE
15. SENÃO
16. TOMAR $(x \in [0, 1])$;
17. SE $(x < e^{\frac{-\Delta}{T}}) S \leftarrow S'$; FIM SE
18. FIM SE
19. FIM ENQUANTO
20. $T \leftarrow \alpha * T$; $IterT \leftarrow 0$;
21. FIM ENQUANTO
22. $S \leftarrow S^*$;
23. RETORNAR (S).

Figura 4.2: *Algoritmo modificado - Simulated Annealing.*

A seguir são apresentadas as heurísticas de distribuição (Figura 4.3) e de programação (Figura 4.4), conforme Mauri (2008)[28].

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, $j = 1, 2, \ldots, n$) FAÇA
5. SELECIONAR (um berço i, $i = 1, 2, \ldots, m$);
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 PARA;

Figura 4.3: *Heurística de Distribuição.*

1. SEJA (k um berço qualquer);
2. PARA (cada navio i atribuído a k) FAÇA
3. $T_i^k = \begin{cases} max(a_i, s^k) & i = 1 \\ max(a_i, T_{i-1}^k + t_{i-1}^k) & i > 1 \end{cases}$
4. FIM PARA;
5. CALCULAR (a função objetivo para o berço k);

Figura 4.4: *Heurística de Programação.*

Detalhes sobre o algoritmo implementado podem ser vistos em Rodrigues et al. (2012)[41].

4.2 Algoritmo Genético

O Algoritmo Genético (AG) como um método aproximado para resolução de problemas de otimização combinatória, foi introduzido em 1975 através do trabalho "Adaptation in Natural and Artificial Systems" escrito por John Holland (1975) [20]. Algoritmos Genéticos são inspirados no princípio Darwiniano da evolução das espécies e na genética. Basicamente, o que um AG faz é criar uma população de possíveis respostas para o problema a ser tratado (inicialização) para depois submetê-la ao processo de evolução. Durante o curso da evolução, populações naturais evoluem de acordo com os princípios de seleção natural e "sobrevivência do mais adaptado". Indivíduos que são mais bem sucedidos na adaptação ao seu ambiente terão melhor chance de sobreviver e se reproduzir, enquanto indivíduos que são menos apropriados serão eliminados. As combinações de boas características de ancestrais, altamente adaptados, deverão produzir uma prole ainda mais ajustada. Desta maneira, as espécies evoluem para se tornar mais e mais bem adaptadas ao seu ambiente. Holland no desenvolvimento dessa teoria mostrou seu interesse em desenvolver sistemas artificiais baseado no processo de adaptação de indivíduos em sistemas naturais e não em resolver problemas de otimização.

Conforme Michalewicz (1992) [32], um AG é um método robusto que pode ser aplicado a uma variedade de problemas de otimização combinatorial, pois diferencia-se dos métodos tradicionais de busca direta. Esses algoritmos utilizam escolhas aleatórias como uma ferramenta para guiar a busca em direção a regiões do espaço com prováveis melhorias, mantendo um conjunto de soluções potenciais (uma população de indivíduos) durante sua execução. Contudo, o modo como um AG faz uso de probabilidade lhe confere uma característica que o distingue de uma mera busca aleatória.

Características importantes que diferenciam outros métodos tradicionais de busca de otimização dos AGs podem ser listadas a seguir:

a) Trabalham com uma codificação do conjunto de parâmetros e não com os próprios parâmetros;

b) Trabalham com uma população e não com um único ponto;

c) Utilizam informações de custo e não derivadas ou outro conhecimento auxiliar;

d) Utilizam regras de transição probabilísticas e não determinísticas.

Segundo Lobo (2005) [24], os algoritmos genéticos utilizam o mesmo vocabulário utilizado na genética, e para uma melhor compreensão de alguns termos, são apresentadas as seguintes definições:

Cromossomo (genótipo): Cadeia de caracteres, representando alguma informação relativa às variáveis do problema. Cada cromossomo representa deste modo, uma solução do problema (indivíduo).

Gen (gene): É a unidade básica do cromossomo. Cada cromossomo tem certo número de gens, cada um descrevendo certa variável do problema.

População: Conjunto de cromossomos ou soluções (indivíduo).

Fenótipo: Cromossomo decodificado.

Geração: Cada iteração que o AG executa para gerar uma nova população.

Operações genéticas: São operações que o AG realiza sobre cada um dos cromossomos, usando operadores genéticos, como o cruzamento e a mutação.

Espaço de busca (região viável): É a região que compreende as soluções possíveis ou viáveis do problema a ser otimizado. Deve ser caracterizado pelas funções de restrições, que definem as soluções viáveis do problema a ser resolvido.

Função objetivo (função aptidão): Construída a partir dos parâmetros envolvidos no problema. Fornece uma medida da proximidade da solução em relação a um conjunto de parâmetros. A função de aptidão permite o cálculo da aptidão de cada indivíduo e fornecerá o valor a ser usado para o cálculo de sua probabilidade de ser selecionado para reprodução.

Aptidão bruta: saída gerada pela função de aptidão para um indivíduo da população.

Aptidão máxima: melhor indivíduo da população.

Mercado (2001) [29], especifica o AG como um processo iterativo que mantém uma população de estruturas (indivíduos, cromossomos), que representam possíveis soluções de um determinado problema. Num Algoritmo Genético, cada indivíduo da população contém a codificação de uma possível solução do problema. A forma como as soluções são codificadas varia em função das características do problema em questão, sendo normalmente usada a codificação binária. Nos algoritmos genéticos com codificação binária, os indivíduos são representados por vetores de dígitos binários, onde cada elemento de um vetor denota a presença (1) ou ausência (0) de um determinado atributo. Cada um desses atributos é conhecido como gene. Os possíveis valores que um determinado gene pode assumir são denominados alelos.

Segundo Dias (2005) [14], na iteração k, uma nova população P_k é criada através de uma fase de reprodução de alguns indivíduos da população P_{k-1}. Esta fase de

reprodução consiste na seleção de indivíduos da geração P_{k-1} para operações de cruzamento e/ou mutação. A seleção de indivíduos pode ser aleatória ou baseada no valor da função aptidão que avalia a qualidade de cada indivíduo enquanto solução do problema.

A Figura 4.5 apresenta um Pseudocódigo para o AG de acordo com Dias (2005) [14]:

```
procedimento AlgoritmoGenético( ){
        X ⟵ X^0
        P^0 ⟵ gerar população inicial
        Avaliar aptidões dos indivíduos de P^0
        enquanto (critério de parada não for satisfeito){
                k ⟵ k + 1
                P^k ⟵ gerar população a partir de P^(k-1)
                Avaliar aptidões dos indivíduos de P^k
                P^k ⟵ selecionar população sobrevivente
        }
        X ⟵ selecionar a melhor solução de P^k
        retornar X
}
```

Figura 4.5: *Pseudocódigo do Algoritmo Genético.*

Os Algoritmos Genéticos representam uma categoria de técnica de otimização que prioriza a diversificação, sendo guiada por um processo que em parte é aleatório, buscando alternativas mais diversificadas no espaço de solução. De uma forma geral, este tipo de algoritmo consegue adquirir informações sobre determinadas características de boas soluções testadas durante o processo de otimização e transmití-las assim para próximas iterações buscando a melhora da qualidade das soluções geradas a cada iteração, (Nogueira, 2009) [35].

4.2.1 Descrição do AG

O Algoritmo Genético aplicado ao Problema de Alocação em Berços tenta simular o processo natural de evolução das espécies como apresentado na Figura 4.6.

Seja uma função $f : \varphi \rightarrow R$ positiva (chamada função objetivo) da qual deseja-se encontrar seu(s) ponto(s) de ótimo (mínimo caso o objetivo seja minimizar). Inicialmente é gerada uma população, sobre a qual serão aplicadas as ações dos passos subsequentes do processo.

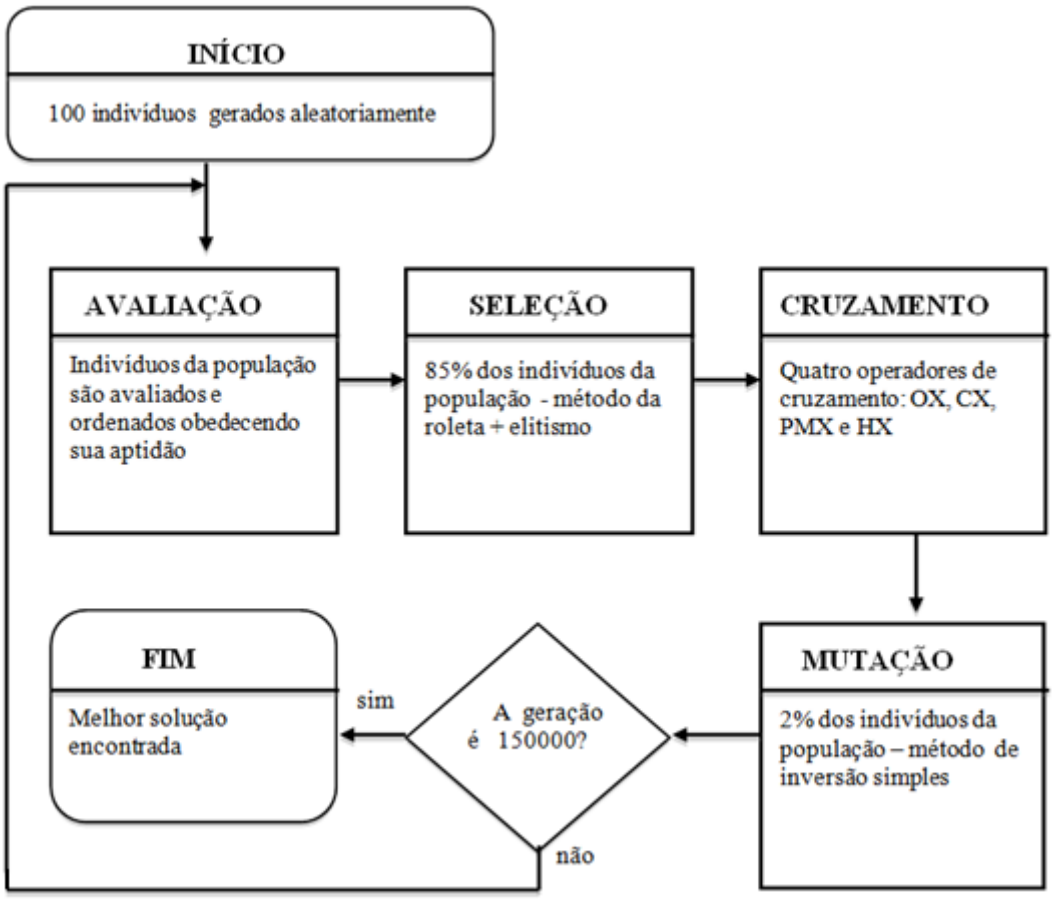


Figura 4.6: *Fluxograma para execução do AG.*

Geração da População Inicial

Assim como no SA, a população inicial para o PAB é gerada a partir da heurística de distribuição (Figura 4.3) de navios.

De acordo com Linden (2006) [26], o tamanho da população a ser utilizado deverá estar entre 50 e 100 indivíduos, pois dependendo do tamanho da população, o desempenho do algoritmo poderá ser afetado prejudicando sua eficiência. Populações muito pequenas possuem baixa diversidade necessária para convergir para uma boa solução, podendo haver uma convergência prematura. Populações muito grandes podem prejudicar o custo computacional do problema, pois o espaço de busca poderá torna-se muito grande.

Existem algumas formas de representação dos indivíduos da população. Em Silva (2007) [45] é apresentada uma forma de representação para o PAB. Nesse trabalho, optou-se pela representação por caminhos, onde o indivíduo é representado por um vetor numérico, tal que a sequência representa os navios N_i a serem alocados em um determinado berço B_k, conforme mostrado na Figura 4.7.

Figura 4.7: *Representação do cromossomo.*

Dessa forma, cada indivíduo (cromossomo) determina uma solução para o problema PAB que são as alocações dos navios aos berços. O custo dessa programação é o valor da função objetivo, ou aptidão do indivíduo da população.

Avaliação dos Indivíduos

Nessa etapa é realizada a mesma heurística de programação (Figura 4.4) feita para o SA, com o objetivo de encontrar a programação de navios de menor custo, função objetivo ou aptidão (que minimiza a função). Para cada indivíduo é então calculado o valor da função objetivo e os melhores indivíduos são escolhidos para reproduzir na geração corrente. Os indivíduos são ordenados de acordo com seu grau de aptidão, ou seja, seu nível de adaptação em relação à população a qual ele pertence.

Segundo Whitley (1994) [47], a execução da função de aptidão nos AGs pode necessitar de quantidade de tempo considerável tendo em vista que, a cada iteração, é necessário avaliar uma população inteira de soluções potenciais (os indivíduos), e não apenas uma solução, como acontece em outras técnicas de otimização.

Seleção

O processo de seleção dos indivíduos da população para a reprodução é feita a partir da avaliação da função aptidão dos indivíduos. Os indivíduos mais aptos constituem uma população intermediária sobre os quais serão aplicados os operadores de cruzamento.

Beasley et al.(1993) [2] afirmam que o processo de seleção influencia muito no comportamento de um AG, pois o tipo de seleção utilizado pode produzir problemas como a convergência prematura para ótimos locais distantes de um ótimo global ou o caso oposto, onde a convergência é muito lenta. Assim não existe um método considerado absoluto, há necessidade de escolhas apropriadas para tratar classes de problemas.

O operador de seleção utilizado foi o método da roleta, no qual os indivíduos a serem selecionados são representados em uma roleta proporcionalmente a sua aptidão. Neste método indivíduos são escolhidos para fazer parte da população intermediária, através de sorteio da roleta. O grau de aptidão de cada indivíduo está relacionado com a porção ocupada pelo indivíduo na roleta, ou seja, indivíduos com aptidão

maior terão uma porção maior na roleta. A roleta é girada um determinado número de vezes, o que depende do tamanho da população. A cada giro da roleta, um indivíduo é apontado pela seta e selecionado, (Ferreira et al., 2003) [15].

O método da roleta, segue os seguintes passos:

1. Some a aptidão de todos os membros da população (A_r);
2. Gere um numero aleatório $n : 0 < n \leq A_r$;
3. Selecione o primeiro indivíduo da população cuja aptidão, somada à aptidão dos indivíduos precedentes é maior ou igual a n, ou seja, $\sum A_i \geq n$.

Uma técnica comumente usada em AGs é a seleção por elitismo, onde alguns dos melhores indivíduos são mantidos a cada geração. Nesse trabalho usou-se também a seleção elitista, ou seja, o indivíduo mais apto de uma população passa automaticamente para a população seguinte. Mais detalhes ver Neto (2010)[34].

Operadores de Cruzamento

Segundo Goldbarg (1989) [18], o operador de cruzamento é a principal força direcionadora em um AG. O operador realiza a troca de partes de pares de indivíduos com o objetivo de tentar obter indivíduos melhores a partir dos indivíduos selecionados. Dessa forma, o principal objetivo do cruzamento é utilizar o conhecimento obtido em pontos do espaço de busca visitados previamente. A aplicação do operador de cruzamento a um par de indivíduos normalmente está sujeita a uma taxa de probabilidade de aplicação, definida como parâmetro para a execução do AG. No trabalho de Linden (2006) [26], são encontrados maiores detalhes sobre a taxa de probabilidade. A taxa de cruzamento indica a probabilidade de aplicação do operador de cruzamento sobre duas soluções selecionadas na roleta. De acordo com Goldbarg (1989) [18], o valor desta probabilidade está compreendido entre 40% a 95%. No algoritmo implementado, a taxa de cruzamento foi fixada em 85% para todos os operadores utilizados.

Os algoritmos genéticos sem os operadores de cruzamento perderiam o sentido, pois eles são os responsáveis por manter a diversidade de indivíduos em uma população, possibilitando a troca aleatoriamente das características genéticas fugindo de regiões de mínimos locais. Existem vários métodos para a realização de um cruzamento entre os indivíduos e as regras que definem como ocorrerá essa troca são específicas para cada caso.

O operador de cruzamento utiliza dois indivíduos de uma população realizando cortes em seus cromossomos e recombinando partes desses cortes na formação de um novo indivíduo. Nesse trabalho, para se adequar a representação dos indivíduos da população por caminhos, selecionamos os principais operadores, que segundo Potvin

(1996) [38] são classificados em operadores que preservam a posição absoluta como o Cruzamento de Mapeamento Parcial (PMX) e Cruzamento por Ciclo (CX) e o operador que preserva a ordem relativa como Cruzamento de Ordem (OX). Além desses operadores de cruzamento que foram implementados, também foi utilizada uma versão heurística de cruzamento (HX). A seguir descreve-se como cada um desses operadores manipula uma solução do PAB.

Muitos problemas para os quais a heurística AG pode ser utilizada apresentam um problema típico em relação à operação de cruzamento, onde elementos de um cromossomo podem aparecer em uma ordem diferente, mas devem ser o mesmo conjunto de elementos. No problema de alocação isso acontece, pois embora a ordem dos atendimentos dos navios possa variar todos os navios precisam ser atendidos. A recombinação em dois pontos seleciona duas localizações para que os cromossomos pais possam ser cortados. Todo o trecho entre estes dois pontos é trocado resultando em dois organismos filhos diferentes. Nesse caso, os filhos gerados por cada conjunto de pais podem resultar, muitas vezes, filhos com navios duplicados enquanto que alguns navios podem não aparecer na lista. A Figura 4.8 mostra essa situação.

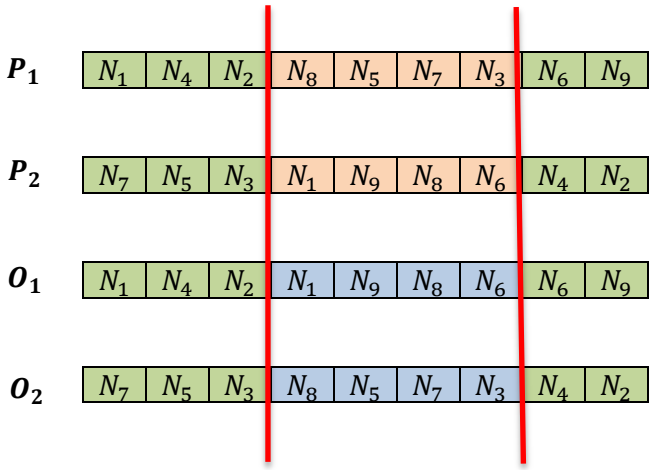


Figura 4.8: *Cruzamento multiponto - AG.*

No exemplo acima, o filho O_1 carece dos navios N_3, N_5 e N_7, tendo os navios N_1, N_6, e N_9 duplicados. Da mesma forma O_2 possui duplicados os navios N_3, N_5 e N_7 e não constam os navios carece dos navios N_1, N_6, e N_9.

Cruzamento de Mapeamento Parcial (PMX)- *Partially-Mapped Crossover*

Para resolver esse problema, outros tipos de cruzamentos foram estudados, um deles é chamado Cruzamento de Mapeamento Parcial (PMX), apresentado a seguir.

O PMX é construído escolhendo uma subsequência de uma turnê de um dos pais preservando a ordem e a posição de tantas posições quanto possíveis do outro pai.

Em primeiro lugar, uma subsequência é delimitada por dois pontos de corte. Depois disso, os segmentos entre os pontos de corte são trocados. Extrai-se do segmento que foi trocado uma série de mapeamentos. No exemplo apresentado: $N_1 \leftrightarrow N_8$, $N_9 \leftrightarrow N_5$, $N_8 \leftrightarrow N_7$ e $N_6 \leftrightarrow N_3$. O segundo passo é preencher as posições para as quais não há conflito, no exemplo mostrado na Figura 4.9, as posições N_4 e N_2 em ambos os cromossomas. O terceiro e último passo é usar os mapeamentos para preencher as posições restantes, antes marcadas com X. Usando o mapeamento $N_1 \leftrightarrow N_8$ no vetor O_1 ocorre um conflito novamente de modo que se deve prosseguir usando o próximo mapeamento $N_8 \leftrightarrow N_7$ a fim de se obter um gene não repetido. As posições em O_1 e O_2 relativas aos elementos duplicados são mostradas na Figura 4.9:

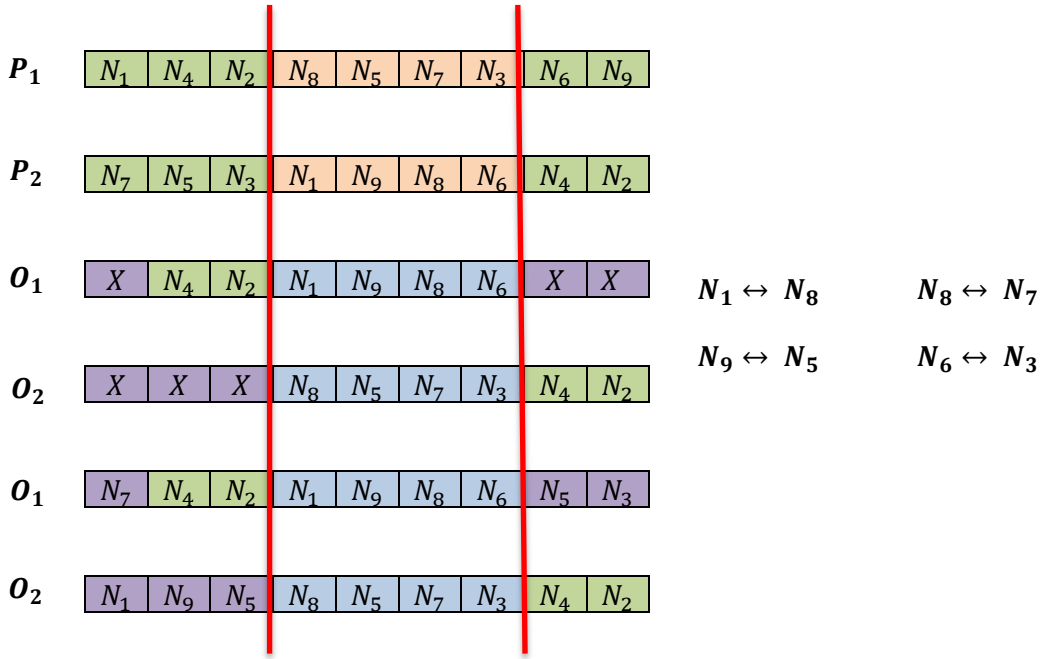


Figura 4.9: *Cruzamento de Mapeamento Parcial - PMX.*

Cruzamento de Ordem (OX) - *Order Crossover*

O OX constrói uma prole escolhendo uma subsequência de uma turnê de um dos pais preservando a ordem relativa das posições do outro pai. Usando o mesmo exemplo do crossover anterior, corta-se a sequência em dois pontos. A partir do segundo ponto de um dos pais as posições do outro pai são copiadas na mesma ordem, desconsiderando-se os símbolos que já estão presentes. Atingindo o fim da cadeia, se prossegue a partir do início da cadeia. No exemplo apresentado na Figura 4.10, preenche-se o cromossomo O_2 seguindo a sequência $N_6 \rightarrow N_9 \rightarrow N_1 \rightarrow N_4 \rightarrow N_2 \rightarrow$

$N_8 \rightarrow N_5 \rightarrow N_7 \rightarrow N_3$ nas posições ainda não estão presentes. São preenchidas as posições N_4, N_2, N_5, N_7 e N_3. O mesmo procedimento é tomado para o cromossomo O_1 seguindo a lista $N_4 \rightarrow N_2 \rightarrow N_7 \rightarrow N_5 \rightarrow N_3 \rightarrow N_1 \rightarrow N_9 \rightarrow N_8 \rightarrow N_6$. Assim, são preenchidas as posições: N_4, N_2, N_1, N_9 e N_6.

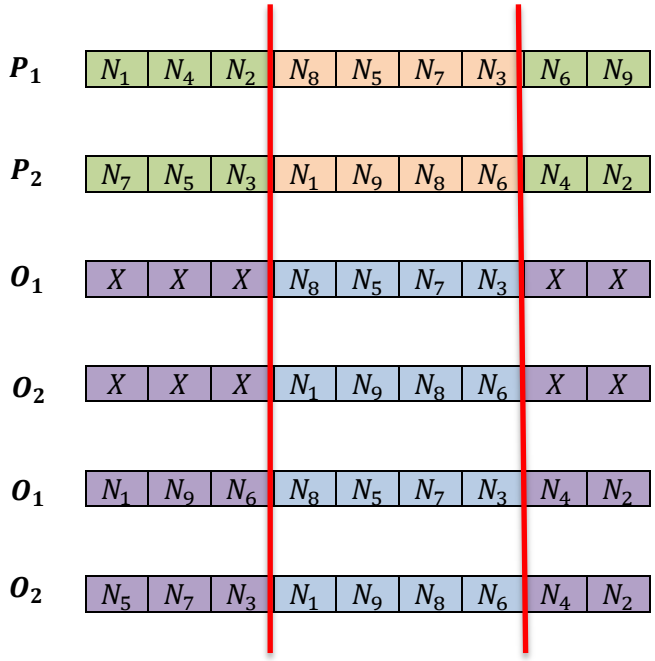


Figura 4.10: *Cruzamento de Ordem - OX.*

Cruzamento por Ciclo (CX)- *Cycle Crossover*

O operador CX preserva a posição absoluta dos navios nos cromossomos pais. O primeiro filho é obtido repetindo o primeiro navio de P_1, procura-se em P_2 a posição correspondente a esse navio e este será herdado preservando a posição que ocupa em P_1, o procedimento segue até que o cromossomo estiver preenchido. Caso seja encontrado um ciclo, as posições seguintes serão ocupadas por indivíduos de P_2.

A Figura 4.11 mostra o esquema para o CX.

O primeiro navio de P_1 foi repetido para O_1, o correspondente em P_2 é o navio N_7 e este é copiado para O_1 preservando a posição que ocupa em P_1. O mesmo ocorre para N_8, N_4 e N_5 quando o navio correspondente em P_2 fecha um ciclo, a partir de então o filho O_1 é preenchido com os indivíduos de P_2 preservando a ordem que ocupam no cromossomo.

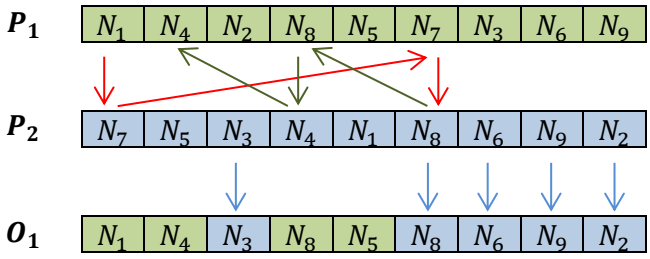


Figura 4.11: *Cruzamento por Ciclo - CX.*

Cruzamento Heurístico (HX)- *Heuristic Crossover*

Este operador utiliza informações heurísticas não exploradas por outros operadores, além de utilizar representação por caminho e considerar a função objetivo entre os navios, ou seja, o menor custo de alocação.

O operador HX pode ser descrito da nos seguintes passos:

1. Escolher um navio aleatório de um dos cromossomos pais;
2. Comparar os custos de alocação para o próximo navios em ambos os pais e selecionar o menor;
3. Se o menor custo escolhido formar um ciclo na rota parcial, então escolha um custo aleatório que não introduza um ciclo;
4. Repetir os passos "1" e "3" até que todos os navios estejam incluídos no berço.

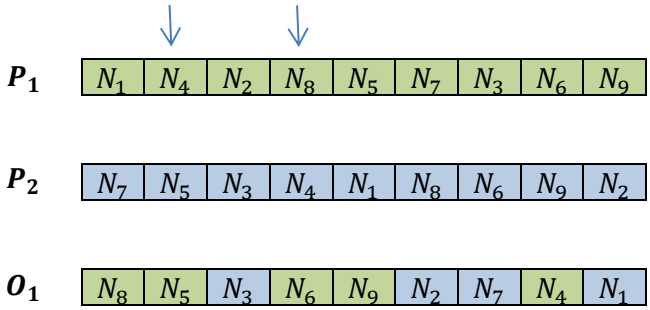


Figura 4.12: *Cruzamento Heurístico - HX.*

Na Figura 4.12 tem-se um exemplo do cruzamento HX, onde foi escolhido aleatoriamente o navio N_8, e dessa forma é analisado o custo de alocação para N_5 e N_6, que são respectivamente os próximos navios em P_1 e P_2. Supondo que o menor custo seja para a sequência $N_8 \rightarrow N_5$, segue-se com a análise de custo para a próxima

alocação. Quando chega-se ao navio N_7 tem-se um ciclo, dessa forma escolhe-se aleatoriamente qualquer navio que não introduza um ciclo. No exemplo foi escolhido o navio N_4 restando apenas o navio N_1 para fechar o cromossomo.

Mutação

A partir de indivíduos selecionados, o operador de mutação faz trocas aleatórias de alguns genes dos indivíduos, fazendo com que dessa forma novas áreas no espaço de busca possam ser pesquisadas. Dessa forma evita a convergência prematura da população, mantendo a diversidade populacional e fugindo de mínimos locais.

Segundo Beasley et al.(1993) [2], a definição dos operadores de cruzamento e mutação, bem como suas taxas de aplicação, pode ser determinante para a ocorrência da convergência. A taxa de mutação não sendo apropriada pode eliminar a possibilidade de exploração do espaço de busca pelo operador de cruzamento, ficando apenas o operador de mutação responsável pela exploração do espaço de busca. Ainda, de acordo com Beasley et al.(1993) [2], a definição dos operadores de cruzamento e mutação, bem como suas taxas de aplicação, por ser determinante para a ocorrência da convergência. A taxa de mutação não sendo apropriada pode eliminar a possibilidade de exploração do espaço de busca pelo operador de cruzamento, ficando apenas o operador de mutação responsável pela exploração do espaço de busca. Na literatura estudada a taxa de mutação deve ser prevista dentro de intervalo de 1% a 5%.

O operador de mutação implementado foi o de inversão simples, proposto por Michalewicz (1996) [32], onde dois pontos no cromossomo são selecionados aleatoriamente e a subsequência entre estes pontos é invertida, conforme mostrado na Figura 4.13.

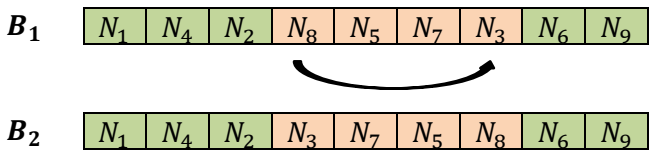


Figura 4.13: *Mutação por Inversão Simples.*

Neste trabalho o operador foi aplicado para cada cromossomo da população com uma probabilidade fixada no valor de 2%.

Critério de Parada

Diferentes critérios podem ser escolhidos para finalizar a execução do AG [24], tais como:

- após um dado número de gerações, ou seja, um total de ciclos de evolução;
- quando a aptidão média ou do melhor indivíduo não melhore mais;
- quando as aptidões dos indivíduos de uma população se tornarem parecidas;
- perda de diversidade da população.

Neste trabalho optou-se como critério de parada o número máximo de gerações, que foi fixada em 150.000.

No Capítulo 5 são apresentados os resultados computacionais obtidos pelas implentações das heurísticas aqui descritas.

Capítulo 5

Resultados

Neste capítulo tem-se a descrição dos testes computacionais executados e os resultados obtidos a partir do cenário proposto para análise.

Os testes computacionais foram realizados em um PC com processador Intel ® Pentium® Dual-Core CPU T4400 2.20 GHz e 4 GB de memória RAM. A implementação foi desenvolvida na linguagem Delphi® baseada em *Object Pascal* (Pascal com extensões orientadas a objetos), pois este permite incluir diversas facilidades de simulação e possibilita a representação de aspectos dinâmicos, dessa forma tornando o modelo mais aderente à realidade que se deseja representar.

5.1 Aplicativo Desenvolvido

Nos últimos anos a inclusão de ferramentas de otimização em simuladores tem-se tornado tendência. Nessas iniciativas observa-se a inclusão de algoritmos aproximativos ou heurísticos, bem como o acoplamento de ferramentas de simulação com ferramentas de otimização, (Cassel e Vaccaro, 2007) [8].

Nesse sentido foi desenvolvido um aplicativo [1] com o intuito de solucionar o PAB utilizando os algoritmos *Simulated Annealing* e Algoritmo Genético para resolver o problema de alocação em berços. Permite ao usuário a simulação de diferentes cenários com o intuito de encontrar a melhor alocação dos navios aos berços e com isso reduzir (minimizar) o tempo de espera em fila dos navios, bem como calcular o melhor horário de atracação a ser oferecido.

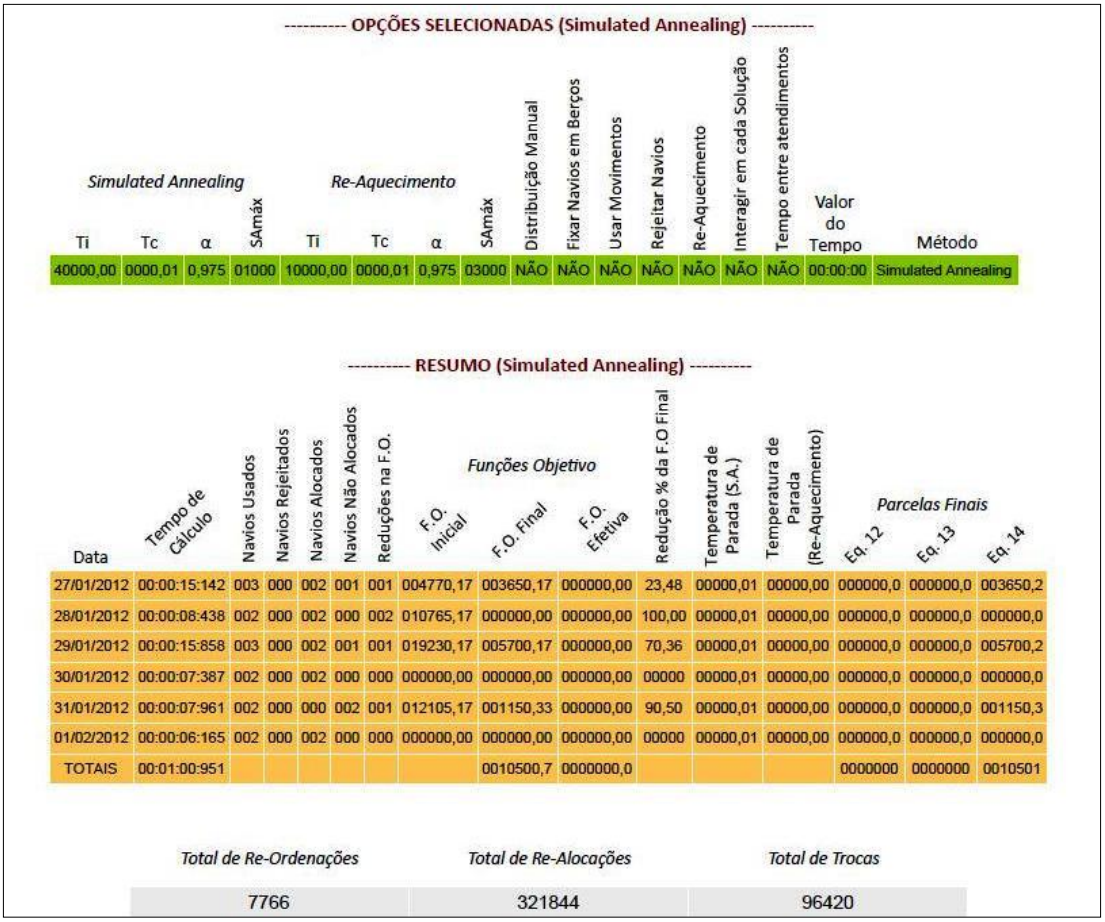
O aplicativo é baseado em informações prévias sobre data e horário de chegada dos navios, duração da janela de tempo (período de tempo em minutos oferecido pelo terminal ao armador, para que o navio num determinado dia possua a garantia de um berço reservado para atracação ou o pagamento de penalidades pelo terminal

[1] O aplicativo foi desenvolvido pelo bolsista Tiago Buchweitz Klug, vinculado ao projeto de Iniciação Científica intitulado Estudos de Algoritmos de Otimização Combinatória para Problemas de Roteamento de Veículos.

caso esse chegue na janela determinada e não possa atracar em virtude de não haver disponibilidade de berço). A partir da informação sobre os horários e datas, o aplicativo distribui e programa os navios aos berços ao longo do período previamente estabelecido.

A utilização do aplicativo oferece certa flexibilidade ao usuário quanto à utilização dos algoritmos. Na simulação por AG pode ser escolhido os operadores de cruzamento, o número de iterações (gerações) e a taxa de mutação. Essas alternativas são responsáveis pelo ajuste dos parâmetros do algoritmo, que têm influência na simulação do problema. Na simulação SA a solução inicial pode ser obtida a partir de quatro maneiras que são: a partir da definição do usuário, troca de movimentos, fixando navios a berços e rejeitando navios, todas essas opções são detalhadas em Rodrigues (2012) [40]. Ainda, o algoritmo SA pode ser executado com ou sem o procedimento chamado Reaquecimento. A técnica de Reaquecimento permite que o usuário tenha a possibilidade de melhorar ainda mais as soluções obtidas, nesse caso o algoritmo é novamente executado, tendo como solução inicial a melhor solução encontrada. O custo operacional de cada dia de alocação dos navios aos berços é informado tanto no AG como no SA permitindo um acompanhamento sobre as soluções obtidas a cada dia de alocação dos navios aos berços.

A interface da ferramenta desenvolvida possibilita gerar um relatório sobre os dados dos navios como a listagem ordenada da chegada dos navios ao porto e listagem dos berços (capacidade, horário de abertura e fechamento). São ainda registrados todos os resultados obtidos durante o processo, as melhores soluções obtidas bem como os parâmetros na aplicação das heurísticas SA e AG. No caso do SA, são também registrados o total de movimentos de re-alocação, re-ordenação e troca de navios permitindo uma avaliação maior sobre essa estrutura de vizinhança que utiliza os três movimentos, onde a escolha é feita de forma aleatória, porém uniformemente distribuída (Figura 5.1). Cabe salientar que o aplicativo oferece a opção de fixar navios aos berços, visto que, em situações reais pode ocorrer de um navio só poder ser alocado em um berço apropriado ao seu tipo de carga.



---------- OPÇÕES SELECIONADAS (Simulated Annealing) ----------

Simulated Annealing				Re-Aquecimento												
Ti	Tc	α	SAmáx	Ti	Tc	α	SAmáx	Distribuição Manual	Fixar Navios em Berços	Usar Movimentos	Rejeitar Navios	Re-Aquecimento	Interagir em cada Solução	Tempo entre atendimentos	Valor do Tempo	Método
40000,00	0000,01	0,975	01000	10000,00	0000,01	0,975	03000	NÃO	NÃO	NÃO	NÃO	NÃO	NÃO	NÃO	00:00:00	Simulated Annealing

---------- RESUMO (Simulated Annealing) ----------

Data	Tempo de Cálculo	Navios Usados	Navios Rejeitados	Navios Alocados	Navios Não Alocados	Reduções na F.O.	Funções Objetivo			Redução % da F.O Final	Temperatura de Parada (S.A.)	Temperatura de Parada (Re-Aquecimento)	Parcelas Finais		
							F.O. Inicial	F.O. Final	F.O. Efetiva				Eq. 12	Eq. 13	Eq. 14
27/01/2012	00:00:15:142	003	000	002	001	001	004770,17	003650,17	000000,00	23,48	00000,01	00000,00	000000,0	000000,0	003650,2
28/01/2012	00:00:08:438	002	000	002	000	002	010765,17	000000,00	000000,00	100,00	00000,01	00000,00	000000,0	000000,0	000000,0
29/01/2012	00:00:15:858	003	000	002	001	001	019230,17	005700,17	000000,00	70,36	00000,01	00000,00	000000,0	000000,0	005700,2
30/01/2012	00:00:07:387	002	000	002	000	000	000000,00	000000,00	000000,00	00000	00000,01	00000,00	000000,0	000000,0	000000,0
31/01/2012	00:00:07:961	002	000	000	002	001	012105,17	001150,33	000000,00	90,50	00000,01	00000,00	000000,0	000000,0	001150,3
01/02/2012	00:00:06:165	002	000	002	000	000	000000,00	000000,00	000000,00	00000	00000,01	00000,00	000000,0	000000,0	000000,0
TOTAIS	00:01:00:951							0010500,7	0000000,0				0000000	0000000	0010501

Total de Re-Ordenações	Total de Re-Alocações	Total de Trocas
7766	321844	96420

Figura 5.1: *Histórico dos resultados*

Segue uma descrição mais detalhada sobre a operacionalização do aplicativo. Os primeiros passos para a execução do algoritmo são adicionar os dados dos navios mencionados anteriormente, em seguida cadastrar os berços e por último escolher a opção de cálculo, onde o usuário pode optar entre as heurísticas SA e AG. Ainda nesta opção podem ser alterados os parâmetros para cada uma das técnicas

Adicionar Navios: informações sobre nome, data e hora de chegada prevista, comprimento m, tempo de atendimento, duração da janela de tempo, custo por minuto (R$) dos navios. A leitura dos dados é feita através de um arquivo Excel® em formato .xls.

Cadastrar Berços: informações sobre nome, capacidade em metros, dia da semana em que o berço opera constando o horário de abertura e fechamento do berço. Um resumo sobre os dias de funcionamento dos berços são fornecidos. Também, existe a possibilidade de leitura dos dados através de um arquivo Excel em formato .xls.

A Figura 5.2 apresenta os navios inseridos ao aplicativo, no calendário é possível visualizar a programação de chegada prevista do conjunto de navios para um determinado dia. A Figura 5.3 apresenta como os berços são cadastrados.

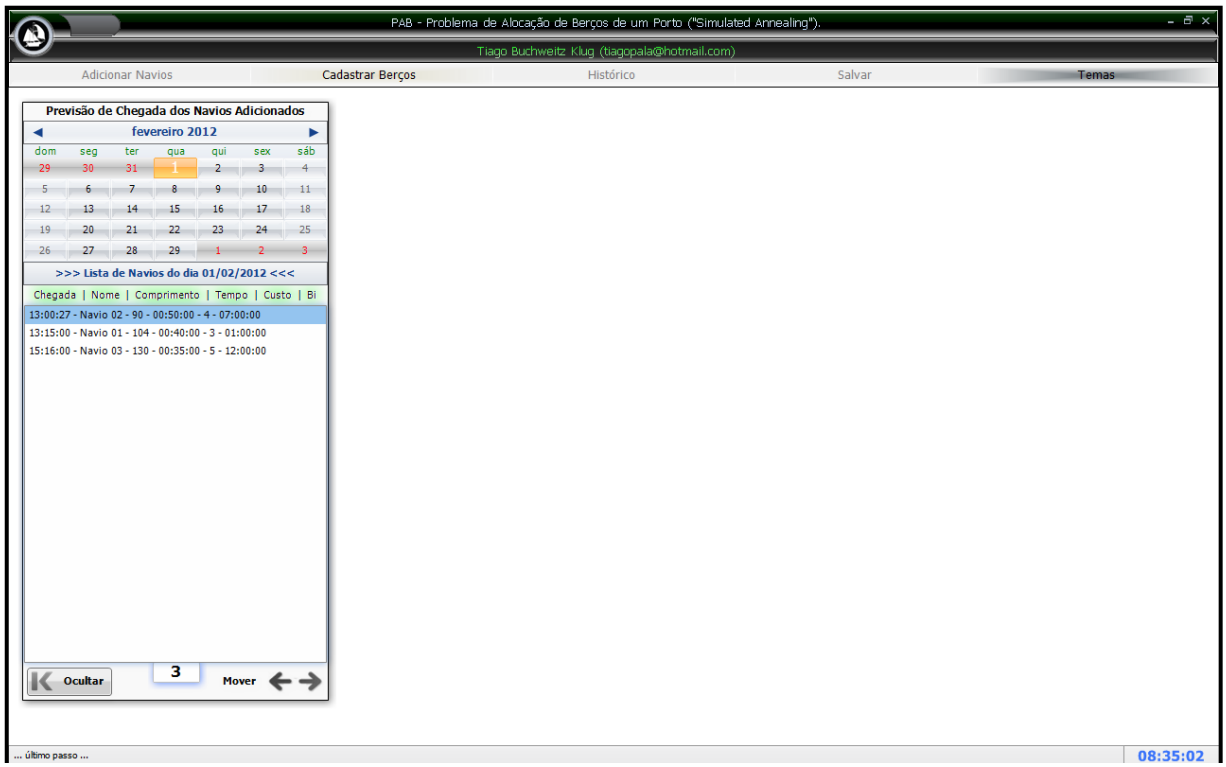


Figura 5.2: *Calendário da programação de chegada dos navios previstos*

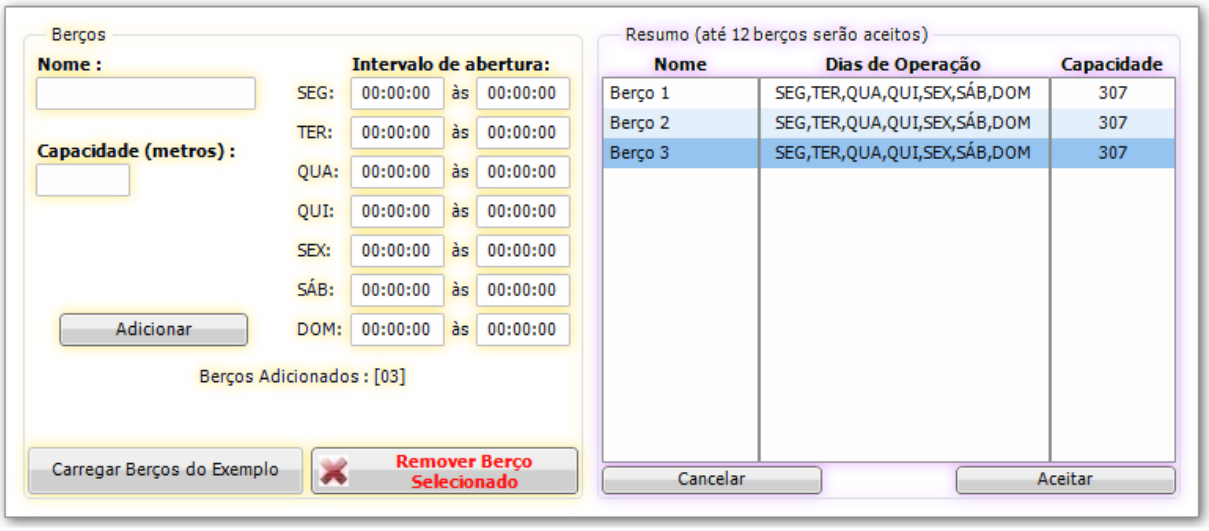


Figura 5.3: *Horário de abertura e fechamento dos berços por dias da semana em um período determinado.*

A Figura 5.4 mostra as opções de cálculo para executar o aplicativo. Estão marcadas todas as opções, pois o aplicativo pode efetuar os cálculos numa única execução ou separadamente a critério do usuário.

Cabe salientar que o aplicativo apresenta para o AG as opções de cruzamento OX, CX, PMX e HX. Foram realizados testes comparativos para cada um desses cruzamentos utilizando o PAB, observou-se que o cruzamento por HX apresentou melhores resultados devido o seu comportamento análogo aos movimentos que são

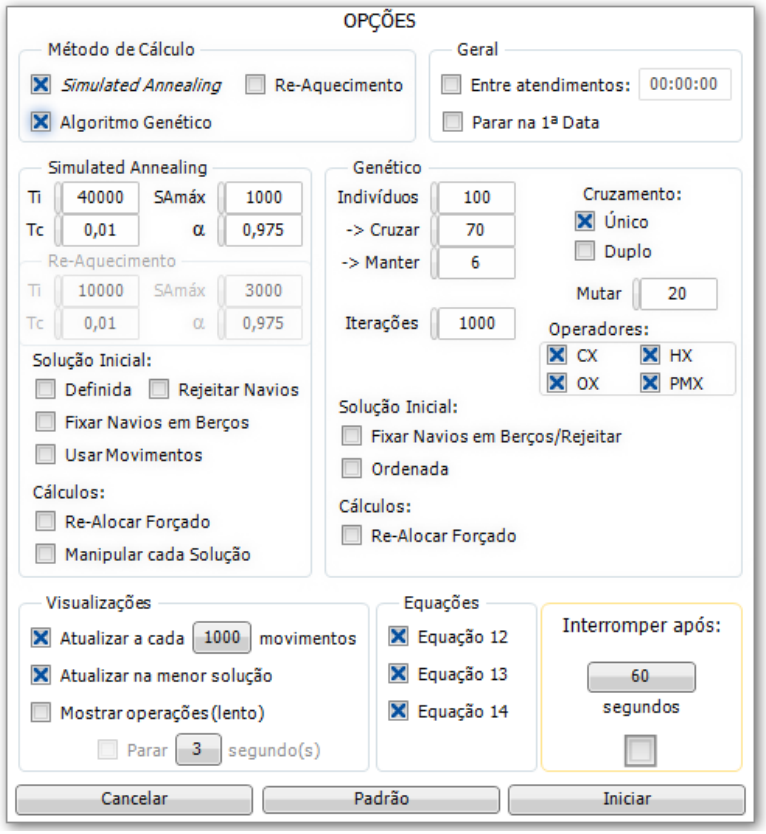


Figura 5.4: *Tela correspondente as opções de cálculo na execução do aplicativo*

feitos pela heurística SA. Portanto, neste trabalho foi utilizado apenas o cruzamento heurístico HX conforme explicado anteriormente.

5.2 Testes Computacionais

O cenário de análise foi constituido a partir de situações reais encontradas na programação de navios de um Terminal de contêineres que utiliza três berços. Essa informações ficam disponíveis no sítio eletrônico do Terminal de Contêineres TECON Rio Grande S/A ($http://www.teconline.com.br/Terminais/Forms/NavioProgramacaoConsultar.aspx$). A partir destes dados foram construidas as planilhas em excel para alimentação do aplicativo utilizado para realizar os testes. Os dados completos podem ser visualizados no Apêndice A.

Os testes foram realizados com base em um terminal de três berços operantes de comprimento fixo e com um total de 55 navios com os seguintes dados: horário e data de chegada prevista, comprimento (m), tempo de atendimento, duração da janela de tempo e valor em moeda corrente de tempo de serviço por minuto. Alguns destes dados são fictícios como é o caso do custo de serviço, pois este não foi fornecido pelo Terminal de Contêineres.

Os parâmetros utilizados para o SA foram:

- $\alpha = 0,975$
- $T_i = 40.000$
- $T_c = 0,01$
- $SA_{max} = 1.000$.

Para o reaquecimento os parâmetros foram $\alpha = 0,975$, $T_i = 10.000$, $T_c = 0,01$ e $SA_{max} = 2.000$. Esses parâmetros foram obtidos de Mauri (2008) [28].

Os parâmetros utilizados para o AG foram:

- tamanho da população de 100 indivíduos
- número de gerações de 150.000 indivíduos
- taxa de cruzamento de 85%
- taxa de mutação fixada em $0,02$.

As penalizações utilizadas para a função objetivo para o modelo reformulado foram $w = [1, 10, 10]$.

A fim de avaliar a potencialidade dos dois algoritmos são apresentados na Tabela 5.1 os resultados de 30 problemas teste, utilizando o modelo Mauri [28]. Na referida tabela constam os valores das funções objetivo final e o tempo de CPU para cada heurística utilizada. Os testes foram realizados separadamente para que fosse possível a contagem do tempo individual de cada heurística.

Testes	Tempo de Execução		F.O. Final	
	SA	*AG*	*SA*	*AG*
x1	28:03:228	19:11:799	192.761,30	185.250,50
x2	29:42:822	22:47:216	197.761,70	192.720,50
x3	32:04:207	28:48:188	189.651,50	184.460,67
x4	32:31:396	30:59:445	192.276,50	184.460,67
x5	29:40:697	21:30:746	197.761,70	185.115,67
x6	28:55:270	20:45:318	198.490,30	184.460,67
x7	29:13:503	15:48:766	198.051,70	185.010,67
x8	27:35:493	17:13:048	189.651,50	185.075,67
x9	26:48:783	19:33:139	200.845,50	184.460,67
x10	30:09:966	21:49:184	189.240,50	185.250,50
x11	30:02:147	22:41:742	192.639,70	184.460,67
x12	29:55:044	22:03:707	197.771,70	184.460,67
x13	27:09:487	19:59:800	200.845,50	187.760,50
x14	29:46:809	22:03:960	192.311,50	186.750,33
x15	29:57:762	19:18:455	192.109,70	187.859,83
x16	29:20:795	20:42:166	192.311,50	186.690,17
x17	26:18:224	21:07:474	190.826,50	185.556,50
x18	30:35:221	20:42:948	192.311,50	184.460,67
x19	27:25:490	22:20:417	194.191,30	189.250,33
x20	27:47:802	22:06:546	195.101,70	184.460,67
x21	28:32:150	18:29:784	195.101,70	187.364,83
x22	29:17:857	18:39:198	192.780,20	184.460,67
x23	28:21:620	18:01:036	192.311,50	185.425,50
x24	28:27:376	20:49:952	189.651,50	184.460,67
x25	30:52:831	21:52:619	195.736,80	188.375,67
x26	27:49:786	18:48:476	195.501,20	185.145,50
x27	28:21:074	18:34:602	195.395,30	184.460,67
x28	25:24:902	17:25:421	190.610,50	189.580,33
x29	34:05:769	18:55:106	195.421,30	184.871,67
x30	27:49:876	21:50:998	192.796,30	188.420,50

Tabela 5.1: *Dados relativos ao tempo e Função Objetivo*

5.3 Resultados Obtidos

Foram realizados 30 testes para cada uma das heurísticas e a partir destes foram coletados os dados da função objetivo final (F.O.) e tempo de execução de cada um. Esses dados estão dispostos na Tabela 5.1. Em uma primeira análise o gráfico apresentado na Figura 5.5 onde foram comparados apenas os valores da (F.O.) para cada teste realizado, notou-se que o AG mostra um comportamento menos variável, pois os valores encontrados para a (F.O.) situam-se entre 185.000 e 190.000, enquanto que o o SA apresenta variações maiores.

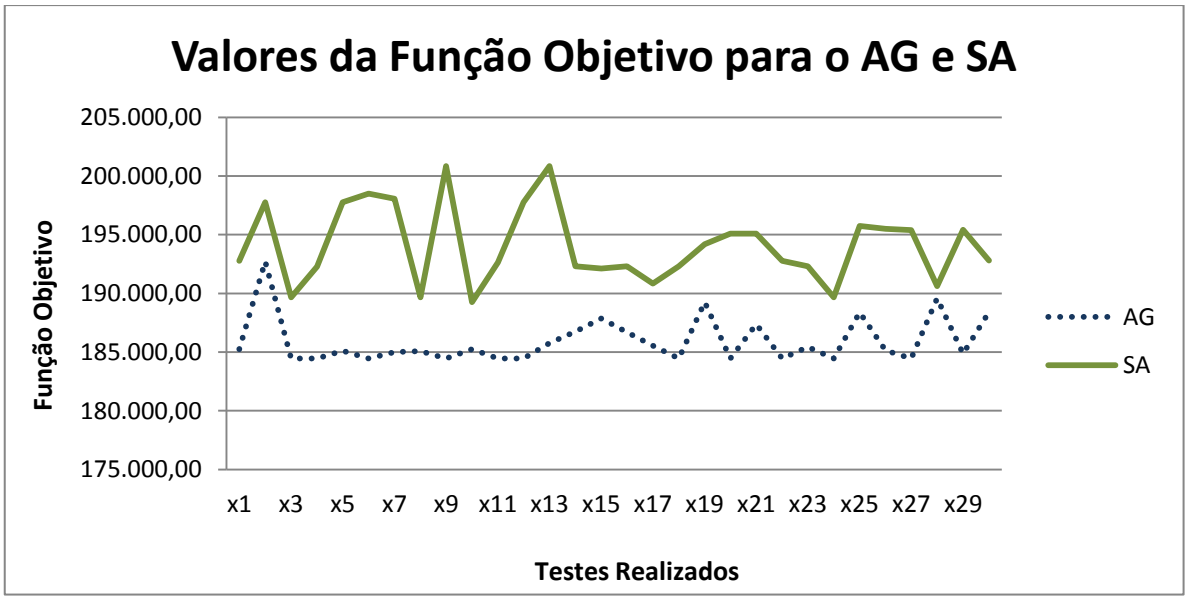


Figura 5.5: *Gráfico comparativo da Função Objetivo final.*

Fazendo uma mesma análise para o tempo de execução de cada heurística obtém-se o gráfico mostrado na Figura 5.6. Observa-se neste, que o AG apresenta um tempo de execução menor que o SA quando comparados os tempos encontrados em cada teste.

Acredita-se que este fato acontece devido a diferença entre as estruturas de vizinhança de cada heurística, pois no SA os movimentos que são realizados são aleatórios e no AG o operador de cruzamento escolhido realiza obrigatoriamente todos os movimentos de troca, reordenação e re-alocação para o problema aqui estudado.

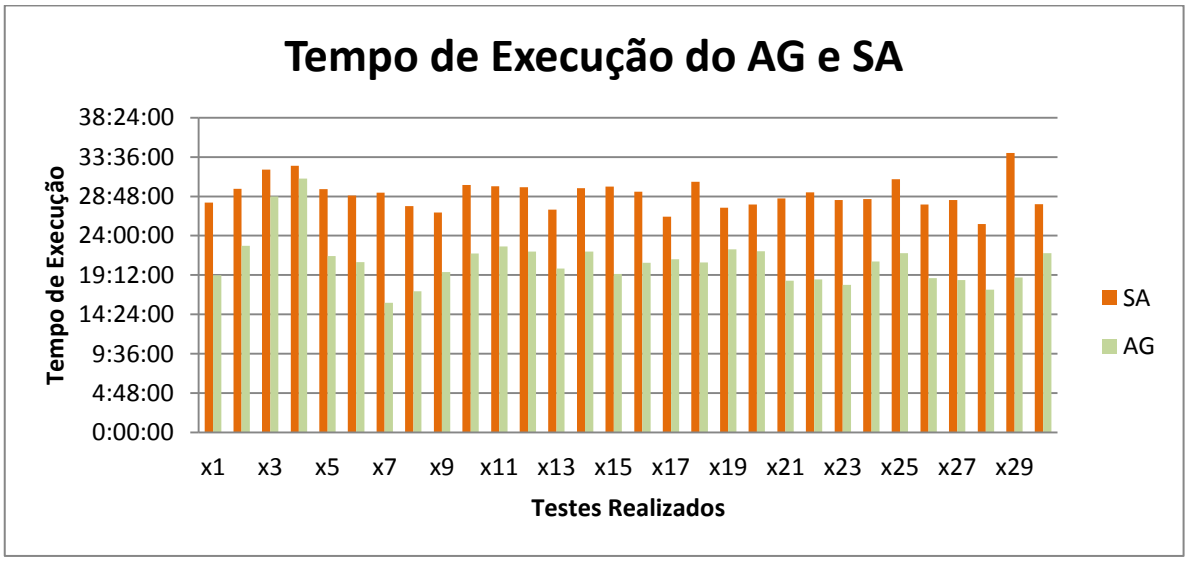


Figura 5.6: *Gráfico comparativo de tempo de execução.*

Tomando-se os dados da Tabela 5.1 como uma amostra de 30 testes, obtém-se as estimativas da média do tempo de execução e função objetivo final, assumindo que estas representam o tempo e função objetivo médio da população, que é desconhecida. Essas e outras estimativas calculadas estão dispostas na Tabela 5.2.

Estimativas	**Heurísticas**	
	Simulated Annealing	**Algoritmo Genético**
Tempo Médio $\overline{x}_t$	29:04:212	20:50:067
Média F.O. final $\overline{x}_f$	194.073,97	185.951,42
Desvio Padrão s	3.267,260	2.023,418
Coeficiente de Variação C_v	1,68351	1,08814

Tabela 5.2: *Estimativas*

Com base nestas estimativas, observou-se que a heurística AG se mostra mais eficiente que a heurística SA, em termos da função objetivo final e tempo de execução, pois o objetivo do Problema de Alocação de Berços é reduzir custo, ou seja, minimizar a função objetivo por meio de ferramentas eficientes em termos de tempo computacional.

Pode-se auferir sobre a consistência da solução através do desvio padrão e do coeficiente de variação calculados, ambos demonstram que os valores para a função objetivo e tempo de execução tendem a estar mais próximos da média para o AG do que para o SA. Este fato pode ser observado nos gráficos de dispersão apresentados nas Figuras 5.7 e 5.8.

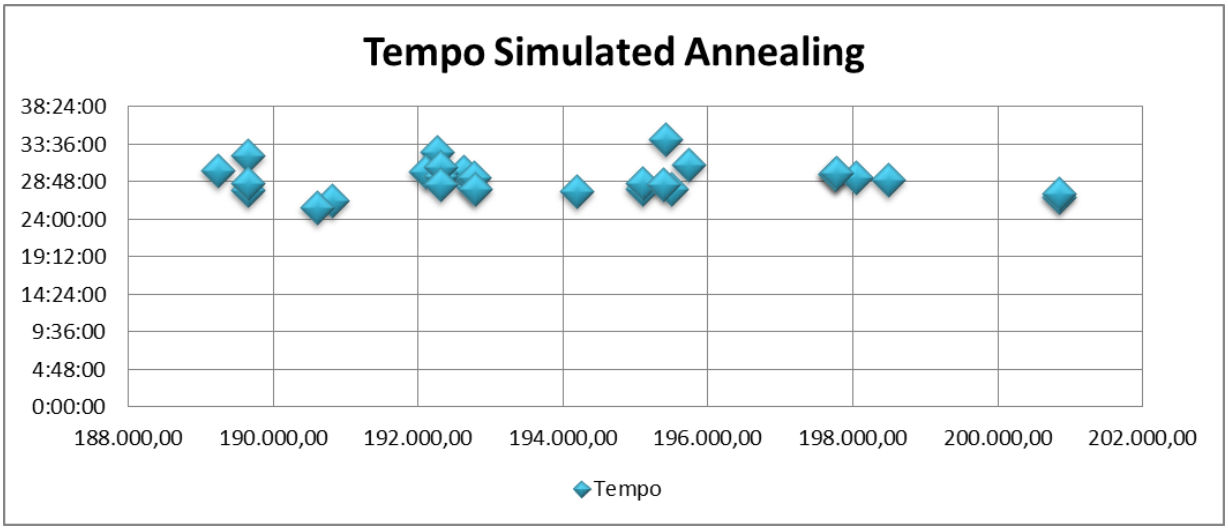


Figura 5.7: *Gráfico de Tempo - SA.*

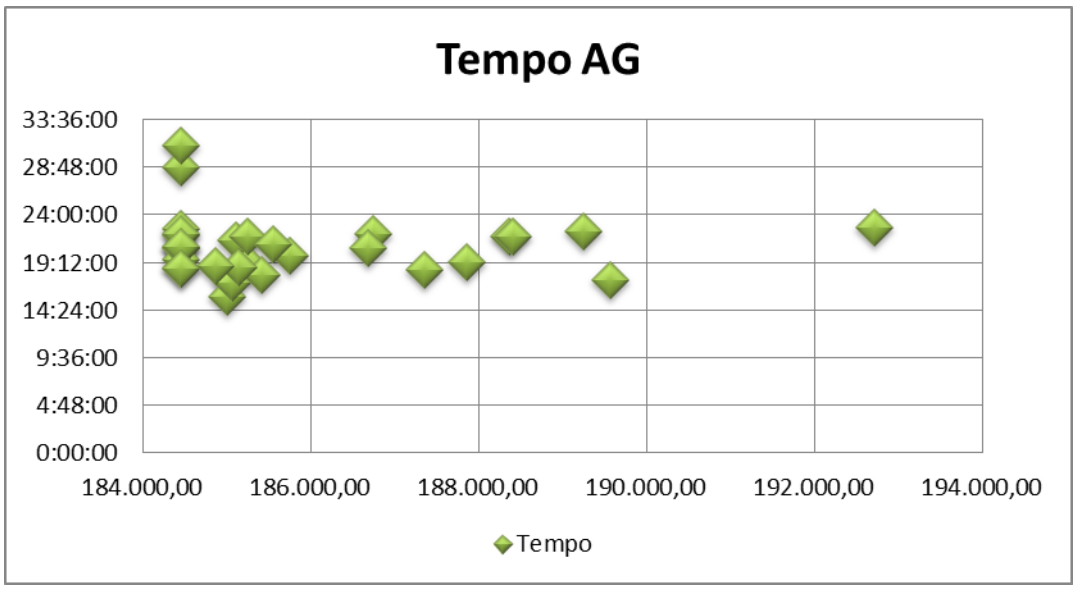


Figura 5.8: *Gráfico de Tempo - AG.*

Outra observação importante foi feita em relação à quantidade de navios não alocados por violação da janela de tempo, em que o Algoritmo Genético deixa de alocar um número menor de navios, ou seja, mais navios são alocados dentro da janela de tempo, o que ocasiona menor custo na função objetivo.

Nesse ponto tem-se um "problema aparente", pois as heurísticas implementadas não consideram a possibilidade de "quebra" no tempo de atendimento por ocasião

do fechamento/abertura de berço, ou seja, todo navio inicia sua operação no dia em que a mesma puder ser concluída. Os navios que não podem concluir a operação no dia da chegada são alocados no dia seguinte na primeira hora disponível após 00:00:000.

Para efeito de trabalho acadêmico as análises aqui tratadas trazem uma discussão a respeito dos resultados obtidos utilizando as heurísticas AG e SA, porém para precisar a eficiência de um ou de outro seria necessária a comparação desses resultados com dados reais, o que infelizmente não se tem acesso. Ainda assim, sob o ponto de vista de um terminal portuário, a heurística AG implementada nessa pesquisa obteve melhores resultados, o que implica uma redução nos custos de operação de um terminal, principal objetivo do PAB.

Um dos objetivos da pesquisa consiste em encontrar instâncias do problema prosposto em que as heurísticas implementadas obtivessem semelhanças e/ou diferenças. Atendendo a esse objetivo foi feita a comparação das programações finais encontradas de acordo com a melhor função objetivo de cada um dos algoritmos. As programações finais para o AG e para o SA estão dispostas, respectivamente, nos Apêndices B.1 e C.1.

Para esta análise foram considerados os navios alocados em cada berço por data, e a função objetivo final de cada dia, esses dados foram retirados do relatório emitido pelo aplicativo. A principal diferença observada foi no valor da função objetivo, que nos dias 12, 13 e 16/01 foi maior no SA, cuja explicação se dá pelo fato da não-alocação de navios.

Data	**Berço 1**	**Berço 2**	**Berço 3**	**F.O./dia**
12/01/2012				
SA	XIN YAN TIAN	AL MARACANA	ER BERLIN	24.289,50
AG	ER BERLIN	XIN YAN TIAN	CAP MELVILLE AL MARACANA	22.179,50
13/01/2012				
SA	CAP MELVILLE MSC GEMMA	LEXA MAERSK	SANTA CLARA	1.335,00
AG	MSC GEMMA	LEXA MAERSK	SANTA CLARA	0,00
16/01/2012				
SA	LOG IN PANT		MSC ADRIATIC MESSOLOGI	14.030,33
AG	MESSOLOGI MSC ADRIATIC	LOG IN PANT		12.695,50

Tabela 5.3: *Diferenças de programação entre AG e SA*

Na Tabela 5.3 tem-se para o dia 12/01 a não-alocação do navio Cap Melville, chegado no dia 11/01 com janela de tempo até 20h do dia 12/01, fato que ocasionou o aumento na F.O. e forçou a violação da janela de tempo deste navio no dia 13/01, adicionando custo na F.O. também para o dia 13/01.

No dia 16/01 ocorre uma situação diferente, o navio Messologi chegado no dia 15/01 com janela de tempo até 21h do dia 16/01 foi alocado no mesmo berço após

o navio MSC Adriatic com chegada no dia 16/01. Observa-se que a aleatoriedade da estrutura de vizinhaça do algoritmo SA pode ter ocasionado tal fato, pois havia um berço disponível para atendimento do navio Messologi evitando que o mesmo violasse sua janela de tempo.

Os testes computacionais apontam a possibilidade de aplicação das heurísticas estudadas em situações reais, ainda que estas representem problemas de médio e grande porte.

Capítulo 6

Conclusões

6.1 Considerações Finais

Os testes realizados nesse trabalho não tiveram a intenção de comparar as técnicas em caráter competitivo, mas sim de mostrar que elas são uma ferramenta a mais na tomada de decisão. O aplicativo desenvolvido para a resolução do problema PAB possibilitará estudos em áreas de pesquisa que objetivam identificar e colecionar um conjunto de instâncias de problemas testes a fim de avaliação da performance dos Algoritmos Genético e *Simulated Annealing*.

Os pesquisadores são muitos cautelosos em generalizar as suas afirmações sobre a performance dos algoritmos, pois ainda não é possível afirmar que uma abordagem algorítmica é melhor do que outra para um problema. No entanto, esta questão pode ficar desprovida de propósito uma vez que muitos estudos atualmente visam hibridizações e tentativas de generalizar as principais características dos algoritmos [42].

DEMESTRESCU E ITALIANO (2000) [13] definem uma nova área de pesquisa, *Experimental Algorithmics* que visa desenvolver sistemas de *software* para suporte à implementação possibilitando uma avaliação empírica dos algoritmos e criando ambientes de testes e simulação. Nesse sentido, deve-se procurar projetar algoritmos para classes de problemas, especificando as características do mesmo que podem ser incluídas no procedimento de busca por soluções de qualidade. Com isso, comparações entre algoritmos, com o intuito de determinar diferentes abordagens para determinadas classes de problemas, tornam-se muito mais relevantes.

Seguem algumas observações e dificuldades encontradas durante a fase de implementação das heurísticas.

- Houve uma preocupação com a adequação dos algoritmos apresentados em relação ao modelo Mauri (2008) [28] e as estratégias utilizadas sobre a recuperação das informações, como por exemplo, a programação final.

- A não disponibilidade de dados utilizados por outros autores, que apenas informam as dimensões do problema, os tempos computacionais e o número de iterações.

- Quanto ao aplicativo, os resultados que são obtidos podem ser acompanhados pela tela, acarretando um tempo de processamento maior dos algoritmos.

- Uma avaliação mais criteriosa seria possível se houvesse condições de aplicar as técnicas estudadas num terminal portuário.

- Tendo em vista que o PAB é um problema de Otimização Combinatória e portanto de natureza não polinomial, deve-se lembrar que o tempo computacional gasto na solução do mesmo não é deterministicamente calculado em função da dimensão do problema.

6.2 Sugestões para Pesquisas Futuras

Esta pesquisa não esgota os estudos sobre o Problema de Alocação de Berços, nem mesmo sobre as heurísticas apresentadas. Os resultados obtidos são encorajadores, porém ficam lacunas a serem preenchidas por estudos posteriores a respeito dos seguintes tópicos:

- Alocação contínua, ou seja, uma reformulação da modelagem do PAB que permite a exclusão do horário de abertura e fechamente de berço, pois na prática alguns portos não operam com essa condição, embora exista a troca de turno de trabalho (estivadores, operadores de máquinas, etc).

- Revisão nas restrições e programação dos algoritmos evitando a "quebra" de tempo de atendimento por ocasião do fechamento/abertura de berço, pois mesmo que haja esse intervalo no atendimento de berços, em situações reais o navio pode começar sua operação imediatamente após sua chegada, bastando que haja berço disponível, independente da pausa no serviço de operação desde que isso não acarrete custos aos terminal, como por exemplo, a violação da janela de tempo do navio.

- Investigação dos movimentos realizados pela estrutura de vizinhança do SA e dos operadores de cruzamento do AG no sentido de melhor aproveitamento da disponibilidade de berços, evitando que um berço atenda mais de um navio enquanto que este poderia ser atendido mais rapidamente por outro berço que esteja disponível.

Apêndice A

Lista de Berços e Lista de Navios

Segue listagem de Berços e de Navios aceitas pelo aplicativo desenvolvido.

Nome do Berço	Intervalo de Abertura (Abertura|Fechamento)														Capacidade (metros)
	SEG		TER		QUA		QUI		SEX		SÁB		DOM		
Berço 1	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	307
Berço 2	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	307
Berço 3	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	00:00:00	23:59:59	307

Figura A.1: *Lista de Berços*

Nome do navio	Hora de chegada prevista	Compr. (m)	Tempo de Atend.	Término da janela de tempo (b_i)	Custo (R$/min)
CAP IRENE	31/01/2012 17:00:00	264	08:15:00	01/02/2012 03:00:00	1
COPACABANA	31/01/2012 15:00:00	179	09:40:00	01/02/2012 01:00:00	1
MSC FORTUNATE	30/01/2012 00:44:00	275	14:15:00	30/01/2012 22:00:00	1
MSC NERISSA	29/01/2012 23:00:00	295	10:30:00	30/01/2012 11:00:00	1
MONTEVIDEO EXPRESS	29/01/2012 08:45:00	277	10:20:00	29/01/2012 20:00:00	1
LOG IN AMAZONIA	29/01/2012 02:00:00	183	13:45:00	29/01/2012 18:00:00	1
HANJIN PIRAEUS	29/01/2012 00:30:00	261	07:55:00	29/01/2012 12:00:00	1
RIO BLANCO	27/01/2012 13:05:00	286	17:00:00	28/01/2012 12:00:00	1
MSC GENEVA	27/01/2012 00:48:00	275	10:30:00	27/01/2012 16:00:00	1
MAERSK LETICIA	26/01/2012 18:00:00	300	14:15:00	27/01/2012 18:00:00	1
LICA MAERSK	26/01/2012 11:45:00	267	05:45:00	26/01/2012 19:00:00	1
CAP ROCA	25/01/2012 19:00:00	234	08:30:00	26/01/2012 06:00:00	1
PUELCHE	25/01/2012 11:00:00	304	13:15:00	26/01/2012 02:00:00	1
ALIANCA IPANEMA	25/01/2012 00:15:00	192	18:10:00	25/01/2012 21:00:00	1
HANSA FLENSBURG	24/01/2012 03:40:00	175	12:10:00	24/01/2012 19:00:00	1
CSAV LAJA	24/01/2012 02:12:00	260	16:15:00	24/01/2012 19:00:00	1
CSAV HOUSTON	23/01/2012 06:00:00	277	09:10:00	23/01/2012 18:00:00	1
MSC TOKYO	23/01/2012 01:24:00	275	14:30:00	23/01/2012 23:00:00	1
MSC ROSARIA	22/01/2012 22:00:00	275	13:25:00	23/01/2012 23:00:00	1
HANSA ATLANTIC	22/01/2012 16:42:00	292	22:00:00	23/01/2012 22:00:00	1
LOG IN JACARANDA	21/01/2012 14:20:00	218	12:30:00	22/01/2012 05:00:00	1
WAN HAI 507	21/01/2012 08:30:00	269	08:30:00	21/01/2012 19:00:00	1
MONTE CERVANTES	21/01/2012 06:00:00	272	06:00:00	21/01/2012 17:00:00	1
MAERSK LIRQUEN	19/01/2012 21:00:00	300	09:45:00	20/01/2012 23:00:00	1
LAURA MAERSK	19/01/2012 13:35:00	266	08:35:00	20/01/2012 07:00:00	1
CAP PALMAS	19/01/2012 10:00:00	208	08:15:00	20/01/2012 03:00:00	1
MAIPO	18/01/2012 17:30:00	306	10:35:00	19/01/2012 12:00:00	1
HANSA AUGSBURG	18/01/2012 00:30:00	176	06:00:00	18/01/2012 20:00:00	1
CAP HENRI	17/01/2012 07:00:00	262	10:30:00	18/01/2012 02:00:00	1
SINGAPORE	16/01/2012 21:00:00	276	09:00:00	17/01/2012 15:00:00	1
RR EUROPA	16/01/2012 14:00:00	201	13:45:00	16/01/2012 12:00:00	1
FOLEGANDROS	16/01/2012 14:00:00	279	10:40:00	17/01/2012 20:00:00	1
MSC ADRIATIC	16/01/2012 01:45:00	277	11:30:00	16/01/2012 18:00:00	1
MESSOLOGI	15/01/2012 22:15:00	294	10:00:00	16/01/2012 21:00:00	1
LOG IN PANTANAL	15/01/2012 10:40:00	183	16:30:00	16/01/2012 08:00:00	1
ZIM SAO PAOLO	14/01/2012 08:30:00	260	08:45:00	14/01/2012 20:00:00	1
RIO DE LA PLATA	14/01/2012 05:45:00	284	11:00:00	14/01/2012 21:00:00	1
MSC GEMMA	12/01/2012 23:30:00	277	10:15:00	13/01/2012 23:00:00	1
SANTA CLARA	12/01/2012 09:36:00	300	13:25:00	13/01/2012 22:00:00	1
ALIANCA MARACANA	12/01/2012 01:00:00	192	11:00:00	12/01/2012 21:00:00	1
XIN YAN TIAN	11/01/2012 23:30:00	280	18:15:00	12/01/2012 22:00:00	1
CAP MELVILLE	11/01/2012 21:00:00	208	12:00:00	12/01/2012 20:00:00	1
LEXA MAERSK	11/01/2012 19:30:00	266	16:20:00	13/01/2012 17:00:00	1
ER BERLIN	11/01/2012 16:40:00	276	16:30:00	12/01/2012 20:00:00	1
HS SMETANA	10/01/2012 23:24:00	176	13:30:00	11/01/2012 23:00:00	1
CAP GREGORY	10/01/2012 02:48:00	264	13:00:00	10/01/2012 21:00:00	1
CCNI AMAZONAS	09/01/2012 13:45:00	275	01:10:00	10/01/2012 04:00:00	1
CAP ISABEL	09/01/2012 07:36:00	264	13:00:00	09/01/2012 23:00:00	1
MSC SARAH	08/01/2012 22:00:00	294	22:00:00	08/01/2012 23:00:00	1
LOG IN SANTOS	08/01/2012 21:50:00	169	06:55:00	09/01/2012 15:00:00	1
MSC ORIANE	08/01/2012 12:30:00	277	12:40:00	09/01/2012 15:00:01	1
HAMMONIA ROMA	07/01/2012 20:42:00	208	16:25:00	08/01/2012 22:00:00	1
RIO MADEIRA	07/01/2012 06:00:00	286	13:00:00	07/01/2012 23:00:00	1
SANTA ROSA	05/01/2012 11:20:00	300	16:00:00	07/01/2012 15:00:00	1
MAULLIN	05/01/2012 10:00:00	306	21:00:00	06/01/2012 22:00:00	1
COPACABANA	05/01/2012 03:30:00	179	09:00:00	05/01/2012 19:00:00	1
LOG IN AMAZONIA	04/01/2012 23:35:35	183	13:00:00	05/01/2012 17:00:00	1
LAUST MAERSK	04/01/2012 21:15:00	267	13:35:00	06/01/2012 02:00:00	1
CAP HARALD	03/01/2012 18:00:00	262	12:05:00	04/01/2012 20:00:00	1
MSC BRINDISI	02/01/2012 12:30:00	275	12:05:00	03/01/2012 15:00:00	1
SCOUT	02/01/2012 08:00:00	93	05:30:00	02/01/2012 20:00:00	1
CHACABUCO	01/01/2012 02:18:00	275	11:30:00	01/01/2012 19:00:00	1

Tabela A.1: *Lista de Navios*

Apêndice B

Programação de Navios com AG

Data	Berço 1	Berço 2	Berço 3	F.O./dia
01/01/2012		CHACABUCO		0,00
02/01/2012	SCOUT			350,17
03/01/2012	MSC BRINDISI			3.650,17
04/01/2012	CAP HARALD			14.056,17
05/01/2012	COPACABANA		LOG IN AMAZONIA	11.125,50
06/01/2012	SANTA ROSA	LAUST MAERSK	MAULLIN	0,00
07/01/2012			RIO MADE	7.870,17
08/01/2012	HAMMONIA ROMA			15.965,50
09/01/2012	MSC ORIANE	MSC SARAH	LOG IN SANTOS	850,00
	CCNI AMAZONAS		CAP ISABEL	
10/01/2012		CAP GREGORY		7.740,17
11/01/2012		HS SMETANIA		33.580,50
12/01/2012	ER BERLIN	XIN YAN TIAN	CAP MELVILLE	22.179,50
			ALIANÇA MARACANA	
13/01/2012	MSC GEMMA	LEXA MAERSK	SANTA CLARA	0,00
14/01/2012	RIO DE LA PLATA		ZIM SÃO PAULO	0,00
15/01/2012				6.850,33
16/01/2012	MESSOLOGI	LOG IN PANTANAL		12.695,50
	MSC ADRIATIC			
17/01/2012	FOLEGANDROS	RR EUROPA	SINGAPORE	750,00
			CAP HENRI	
18/01/2012	HANSA AUGSBURG			2.450,17
19/01/2012	CAP PALMAS	MAIPO		4.565,17
		LAURA MAERSK		
20/01/2012			MAERSK LIRQUEN	0,00
21/01/2012		MONTE CERVANTES	WAN HAI 507	1.700,17
22/01/2012		LOG IN JACARANDA		15.670,33
23/01/2012	HANSA ATLANTIC	MSC TOKIO	MSC ROSARIA	3.745,00
			CSAV HOUSTON	
24/01/2012	CSAV LAJA		HANSA FLENSBURG	0,00
25/01/2012			ALIANÇA IPANEMA	2.250,33
26/01/2012	PUELCHE	CAP ROCA		5.295,17
		LICA MAERSK		
27/01/2012	MSC GENOVA		MAERSK LETICIA	3.650,17
28/01/2012	RIO BLANCO			0,00
29/01/2012	HANJIN PICAEUS		LOG IN AMAZONIA	6.320,17
	MONTEVIDEO EXPRESS			
30/01/2012	MSC NERISSA	MSC FORTUNATI		0,00
31/01/2012				1.150,33
01/02/2012		CAP IRENE	COPACABANA	0,00

Tabela B.1: *Programação de Navios - AG*

Apêndice C

Programação de Navios com SA

Data	Berço 1	Berço 2	Berço 3	F.O./dia
01/01/2012	CHACABUCO			0,00
02/01/2012		SCOUT		350,17
03/01/2012			MSC BRINDISI	3.650,17
04/01/2012	CAP HARALD			14.056,17
05/01/2012	LOG IN AMAZONIA	COPACABANA		11.125,50
06/01/2012	LAUST MAERSK	SANTA ROSA	MAULLIN	0,00
07/01/2012	RIO MADE			7.870,17
08/01/2012	HAMMONIA ROMA			15.965,50
09/01/2012	MSC ORIANE	LOG IN SANTOS	MSC SARAH	850,00
	CCNI AMAZONAS	CAP ISABEL		
10/01/2012	CAP GREGORY			7.740,17
11/01/2012	HS SMETANIA			33.580,50
12/01/2012	XIN YAN TIAN	ALIANÇA MARACANA	ER BERLIN	24.289,50
13/01/2012	CAP MELVILLE	LEXA MAERSK	SANTA CLARA	1.335,00
	MSC GEMMA			
14/01/2012	ZIM SÃO PAULO	RIO DE LA PLATA		0,00
15/01/2012				6.850,33
16/01/2012	LOG IN PANTANAL		MSC ADRIATIC	14.030,33
			MESSOLOGI	
17/01/2012	SINGAPORE	FOLEGANDROS	RR EUROPA	750,00
	CAP HENRI			
18/01/2012	HANSA AUGSBURG			2.450,17
19/01/2012	MAIPO		CAP PALMAS	4.565,17
	LAURA MAERSK			
20/01/2012	MAERSK LIRQUEN			0,00
21/01/2012	WAN HAI 507		MONTE CERVANTES	1.700,17
22/01/2012		LOG IN JACARANDA		15.670,33
23/01/2012	MSC ROSARIA	HANSA ATLANTIC	MSC TOKIO	3.745,00
	CSAV HOUSTON			
24/01/2012	CSAV LAJA		HANSA FLENSBURG	0,00
25/01/2012	ALIANÇA IPANEMA			2.250,33
26/01/2012	CAP ROCA		PUELCHE	5.295,17
	LICA MAERSK			
27/01/2012		MSC GENOVA	MAERSK LETICIA	3.650,17
28/01/2012	RIO BLANCO			0,00
29/01/2012	LOG IN AMAZONIA	HANJIN PICAEUS		6.320,17
		MONTEVIDEO EXPRESS		
30/01/2012	MSC FORTUNATI	MSC NERISSA		0,00
31/01/2012				1.150,33
01/02/2012	CAP IRENE		COPACABANA	0,00

Tabela C.1: *Programação de Navios - SA*

Referências Bibliográficas

[1] ANTAQ. **Agência Nacional de Transportes Aquaviários**. Disponível em: < *http : //www.antaq.gov.br/Portal/Default.asp* > Acesso em: 02 de março de 2012. 14

[2] BEASLEY, D., BULL, D. R. e MARTIN R. R. **An Overview of Genetic Algorithms**: Part I, Fundamentals. University Computing, vol. 15, no. 2, pp. 58-69, 1993. 33, 39

[3] BECCENERI, J. C. **Meta-heurísticas e Otimização Combinatória**: Aplicações em Problemas Ambientais. *Computação e Matemática Aplicada às Ciências e Tecnologias Espaciais*. INPE, São José dos Campos, p.65-81, 2008. x, 11, 12, 23, 25

[4] BLUM, C. ; ROLI, A. **Metaheuristics in Combinatorial Optimisation**: Overview and Conceptual Comparison. Technical Report TR/IRIDIA/2001-13, IRIDIA, Université Libre de Bruxelles, Belgium, 2001. 24

[5] BOTTER, R. C.; PATRÍCIO, M. **Análise de regras de atracação de navios em terminais de conteineres** Disponível em: *http : //www.ipen.org.br/downloads/XIX/CT5_PUERTOS_Y_OBRAS_PORTUARIAS/Marcelo%20Patricio%20%20_2_.pdf* Acesso em: 12 de setembro de 2011. 14, 16

[6] BRASIL, Lei nº 8.630, de 25 de fevereiro de 1993. Dispõe sobre o regime jurídico da exploração de portos organizados e das instalações portuárias e dá outras providências. **Diário Oficial da República Federativa do Brasil**, Brasília, DF, 26 fev. 1993. p. 2351. 13

[7] BURIOL, L. **Algoritmo memético para o Problema do Caixeiro Viajante Simétrico como parte de um Framework para Algoritmos Evolutivos**. Dissertação (Mestrado em Engenharia Elétrica), Unicamp, Campinas, 2000. 10

[8] CASSEL, G. L . e VACCARO, G. L. R. A **Aplicação de simulação-otimização para a definição do mix ótimo de produção de uma indústria metal-mecânica**. Encontro Nacional de Engenharia de Produção, Foz do Iguaçu. 2007. 41

[9] CHAVES, A. A. Disponível em < *http : //www.feg.unesp.br/ chaves /metaheuristicas.html* > Acesso em: 18 de maio de 2010. 24

[10] CORDEAU, J. F.; LAPORTE, G.; MERCIER, A. **A unified tabu search heuristic for vehicle routing problems with time windows**. *Journal of the Operational Research Society*, v.52, 928-936, 2001. 18, 19, 20

[11] CORDEAU, J.F.; LAPORTE, G.; LEGATO, P; MOCCIA, L. **Models and tabu search heuristics fot the berth allocation problem**. *Transportation Science*, v.39, p. 526-538, 2005. 16

[12] CORMEN, T. H.; LEISERSON, C. E.; RIVEST, R. L; Stein, C. **Algoritmos: Teoria e Prática**. Tradução da Segunda Edição Americana por Vandenberd D. de Souza. Rio de Janeiro: Campus, p.763-807, 2002. 10

[13] DEMESTRECU, C. e ITALIANO, G.F. **What Do We Learn from Experimental Algorithmics?** Invited Lecture, Proc. 25th International Symposium on Mathematical Foundations of Computer Science (MFCS 2000), August 28 – September 1, Bratislava, Slovakia, 36–51. 2000 53

[14] DIAS, A. **O problema da p-mediana aplicado ao problema da gestão óptima da diversidade**. Dissertação (Mestrado em Matemática e Aplicações), Universidade de Aveiro. Portugal, 2005. 30, 31

[15] FERREIRA, A. C. P. L; BRAGA, A. P.; LUDERMIR, T.B. **Computação Evolutiva**. In: Rezende,S.O.. Sistemas Inteligentes ? Fundamentos e Aplicações. São Paulo: Manole, 2003. Cap. 9,p.225-246. 34

[16] FOULDS, L. R. **Combinatorial Optimization for Undergraduates**. *Springer-Verlag*, New York, p. 114, 1984. 11

[17] GAREY, M. R. & JOHNSON, D. S. **Computers and Intractability: a guide to the theory of NP-Completeness**. San Francisco, Freeman, 1979. 10

[18] GOLDBARG, M. C.; LUNA, H. P. **Otimização Combinatória e Programação Linear** - Modelos e Algoritmos. Rio de Janeiro, Ed. Campus, 2000. 8, 34

[19] GROTSCHEL, M. e LOVÁSZ, L. e SCHRIJVER, A. **Geometric algorithms and combinatorial optimization**. Springer-Verlag, 1988. 10

[20] HOLLAND, J.H.**Adaptation in Natural and Artificial Systems**. University of Michigan Press, 1975. 29

[21] IMAI, A.; NISHIMURA, E.; PAPADIMITRIOU, S. **The dynamic berth allocation problem for a container port**. *Transportation Research Part B: Methodological*, v. 35, n. 4, p. 401-417, 2001. 1

[22] KIRKPATRICK, S.; GELLAT, D. C.; VECCHI. M.P. **Optimization by Simulated Annealing**, *Science*, v.220, p. 671- 680, 1983. 22, 23

[23] KITZMANN, D. **Ambiente Portuário**. Rio Grande: Editora da FURG, 2010. 13

[24] LOBO, E. L. M. **Uma Solução do Problema de Horário Escolar via Algoritmo Genético Paralelo**. Dissertação de Mestrado. Programa de Pós-Graduação em Modelagem Matemática e Computacional, CEFET, Belo Horizonte, MG, Brasil, 2005. 29, 40

[25] LOESCH, C.; HEIN, N. **Pesquisa Operacional**: Fundamentos e Modelos. Blumenau: Ed. da FURB, 1999. 4

[26] LINDEN, Ricardo. **Algoritmos Genéticos**. Brasport, 2006. 32, 34

[27] MARQUES, F. P.; ARENALES, M. N. (2002). **O Problema da mochila compartimentada e aplicações**. *Pesquisa Operacional, 22(3), 285-304.* Disponível em: *http : //www.scielo.br/scielo.php?pid = S*0101 − 74382002000300001&*script* = *sci*$_a$*rttext* Acesso em 07 de abril de 2012. 7

[28] MAURI, G.R.; OLIVEIRA. A.C.M.; LORENA L.A.N. **Heurística baseada no Simulated Annealing aplicada ao problema de alocação de berços**. *GEPROS - Gestão da Produção, Operações e Sistemas*, Ano 3, v.1, n.1, p. 113-127, 2008. v, vi, x, 16, 17, 20, 26, 27, 46, 53

[29] MERCADO, N. B. G. **Técnica de Busca Baseada em Algoritmo Genético para Localização de p-Medianas**. Dissertação de Mestrado. UFSC, Florianópolis, 2001. 30

[30] Metaheuristics Network. Disponível em: *http : //www.metaheuristics.net /index.php?main* = 1. Acesso em: 06 de fevereiro de 2012. 11

[31] METROPOLIS, N. C. et. al. **Equation of state calculations by fast computing machines**. *Journal of Chemical Physics*, v. 21, 1087-1092, 1953. 23

[32] MICHALEWICZ, Z. **Genetic Algorithms + Data Structures = Evolution Programs**. Springer- Verlag, 1992. 29, 39

[33] LEGATO, P.; MONACO, F.; TIGANI, N. **Berth planning at gioia tauro's maritime terminal by logistic distribution models.** In: *Annual Conference Of Italian Operational Research Society*, 32, 2001, Cagliari. Proceedings Cagliari: AIRO, 2001. 17

[34] NETO, J. C. R. **Modelagem dos Algoritmos Genético Simples e** *Simulated Annealing* **por cadeias de Markov**. Dissertação de mestrado, Programa de Pós-graduação em Matemática Aplicada e Estatística - CCET - UFRN, 2010. 22, 34

[35] NOGUEIRA, P. B. **Metodologia de Otimização Probabilística de estratégias de produção baseada em algoritmos genéticos** Dissertação de Mestrado em Engenharia Mecânica. Universidade Estadual de Campinas. Campinas, 2009. 31

[36] OLIVEIRA, C. T. **Modernização dos portos**. 5. ed. São Paulo : Aduaneiras, 2011. 14

[37] PIZZOLATO, N. D.; GANDOLPHO, A. A. **Técnicas de Otimização**. Rio de Janeiro: LTC, 2009. x, 4, 5

[38] POTVIN, J. Y. **Genetic algorithms for the traveling salesman problem**. Annals of Operations Research 6, p. 339-370, 1996. 35

[39] RAYAWARD-SMITH, V. J. et al. **Modern Heuristic Search Methods**. New York: J.W. and Sons, 1996. 1

[40] RODRIGUES, M. H. P. **Simulated Annealing: Uma proposta de resolução para o Problema de Alocação de Berços em Terminais de Contêiners**. Dissertação de Mestrado, Pós-Graduação de Engenharia Oceânica, Universidade Federal do Rio Grande (2012). 11, 17, 42

[41] RODRIGUES, M. H. P., MACHADO, C.M.S, LIMA, M. L.P. **Simulated Annealing Aplicado ao Problema de Alocação de Berços**. *Journal of Transport Literature*, 2012. 28

[42] SALGADO M. R., ARAÚJO O. C. B. **Revisão bibliográfica sobre estudos comparativos entre algoritmos genéticos, simulated annealing e busca tabu** Universidade Estadual de Campinas. Disponível em ftp : $//ftp.dca.fee.unicamp.br/pub/.../revisao_bibliografica/tema23.doc$. Acesso em: 11 de junho de 2012. 23, 53

[43] SCHWEFEL, H.P. e TAYLOR, L. **Evolution and Optimum Seeking**. United States of America, John Wiley & Sons Inc, pp. 87-88, 1994. 8

[44] SILVA, A. S. N. **Estudo e Implementação, Mediante Recozimento Simulado, do Problema de Alocação de Salas**. Monografia, Departamento de Ciência da Computação, Universidade Federal de Lavras (2005). 11

[45] SILVA, V.M.D.; COELHO A.S. **Uma Visão sobre o problema de alocação de berços**. *Revista Produção Online*, v.7, n.2. ISSN 1676-1901, 2007. 32

[46] SUCUPIRA, I. R. **Métodos heurísticos genéricos**. 2004. 41 f. Monografia (Departamento de Ciência da Computação). Universidade de São Paulo, São Paulo. Disponível em: $http$: $//www.ime.usp.br/\ igorrs/monografias/metahiper.pdf$. Acesso em 06 de fevereiro de 2012. 11

[47] WHITLEY, D. A **Genetic Algorithm Tutorial**. Statistics and Computing, vol. 4, pp. 65-85, 1994. 33