ADRIANO HEIS

UM ALGORITMO EVOLUTIVO PARA O PROBLEMA DE CORTE DE ESTOQUE UNIDIMENSIONAL INTEIRO

MARINGÁ

2007

ADRIANO HEIS

UM ALGORITMO EVOLUTIVO PARA O PROBLEMA DE CORTE DE ESTOQUE UNIDIMENSIONAL INTEIRO

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Estadual de Maringá, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Ademir Aparecido Constantino
Co-orientador: Prof. Dr. Silvio Alexandre de Araújo

MARINGÁ

2007

Dados Internacionais de Catalogação-na-Publicação (CIP)
(Biblioteca Central - UEM, Maringá – PR., Brasil)

Heis, Adriano
H473a Um algoritmo evolutivo para o problema de corte de estoque unidimensional inteiro / Adriano Heis. -- Maringá : [s.n.], 2007.
107 p. : il. , figs., tabs.

Orientador : Prof. Dr. Ademir Aparecido Constantino.
Co-orientador : Prof. Dr. Silvio Alexandre de Araujo.
Dissertação (mestrado) - Universidade Estadual de Maringá. Programa de Pós-graduação em Ciência Computação, 2007.

1.Otimização inteira. 2. Heurísticas. 3. Algorítmo evolutivo. 4. Problema de corte unidimensional inteiro. 5. Inteligência artifícial. 6. Ciência da Computação. I. Universidade Estadual de Maringá. Programa de Pós-graduação em Ciência da Computação.

Cdd 21.ed. 004.0151

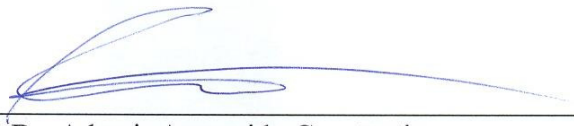
ADRIANO HEIS

UM ALGORITMO EVOLUTIVO PARA O PROBLEMA DE CORTE DE ESTOQUE UNIDIMENSIONAL INTEIRO

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Estadual de Maringá, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

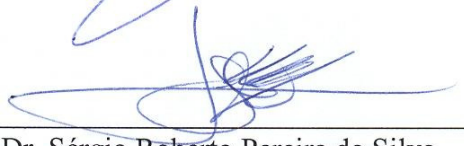
Aprovado em 10/09/2007.

BANCA EXAMINADORA

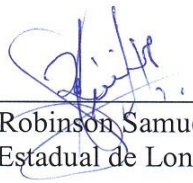


Prof. Dr. Ademir Aparecido Constantino
Universidade Estadual de Maringá – DIN/UEM

Prof. Dr. Silvio Alexandre de Araujo
Universidade Estadual Paulista – DCCE/UNESP



Prof. Dr. Sérgio Roberto Pereira da Silva
Universidade Estadual de Maringá – DIN/UEM



Prof. Dr. Robinson Samuel Vieira Hoto
Universidade Estadual de Londrina – MAT/UEL

DEDICÁTORIA

A meus pais, Ademar (*em memória*) e Inês, meus exemplos e orgulho de chamá-los de pai e mãe.

A minha esposa Renata e nossa filha Gabriela que são as razões da minha vida.

AGRADECIMENTOS

Ao meu orientador Prof. Dr. Ademir Aparecido Constantino, pela dedicação despendida e apoio.

Ao meu co-orientador Prof. Dr.Silvio Alexenadre de Araújo pela dedicação, pela doação dos seus conhecimentos e dias de descanso, pelo empenho em me fazer entender as modelagens matemáticas, e paciência nos seus conhecimentos.

Aos coordenadores do mestrado (professora Itana e professor Sérgio), ao colegiado e todos os professores do mestrado por todo apoio.

A Inês, secretária do mestrado, que sempre esteve de prontidão a ajudar.

A todos os meus novos amigos conhecidos no mestrado, em especial a Adriana Herden, Everson Matias de Mora, Flávio Schiavoni, Jô Sato, José Rafael Carvalho, Lafaiete Henrique Rosa Leme e Rafael Alessandro Gatto.

A Kelly Poldi por estar sempre à disposição em sanar dúvidas, ceder os exemplos utilizados e a interpretar os resultados.

A CAPES, pelo apoio financeiro.

RESUMO

Problemas de corte e empacotamento estão presentes em diversos segmentos da indústria onde existe a necessidade de se cortar peças maiores em peças menores. Dentre os vários tipos de problemas de corte e empacotamento tem-se os problemas de corte de estoque, que consiste de cortar objetos em estoque em itens menores de forma a atender uma determinada demanda, otimizando um certo critério (função de objetivo). No presente trabalho é feita uma revisão bibliográfica de modelos matemáticos e métodos de solução que tratam de problemas de corte de estoque unidimensionais inteiros, ou seja, apenas uma dimensão é considerada e existem variáveis do problema que devem assumir valores inteiros. Um método de solução é proposto, diferente dos métodos encontrados na literatura, para o caso específico em que se têm diferentes tipos de objetos em estoque em quantidade limitada. O método proposto é baseado nos conceitos de meta-heurística utilizando algoritmos evolutivos para a resolução do problema. Testes computacionais foram realizados considerando um conjunto de exemplos gerados aleatoriamente, e os resultados de várias heurísticas foram comparados com os resultados do método proposto. O novo método mostra-se eficiente em relação à qualidade dos resultados obtidos e na redução do tempo computacional.

Palavras-Chave: Otimização inteira, Problema de corte e empacotamento, Algoritmos evolutivos.

ABSTRACT

Cutting and Packing problems arise in different kinds of industries where a set of small items (output, demand) is cut from a set of large objects (input, supply). Among the different types of cutting and packing problems is the cutting stock problem, which consists of cutting a set of available object in stock in order to produce ordered smaller items in such a way as to optimize a given objective function. In this work we present a review of mathematical models and solution methods dealing with the one-dimensional integer cutting stock problem, e.g., only one-dimension is considered and there are integer variables. A solution method is proposed, different from the method found on the literature, to the specific case in which there are several stock lengths available in limited quantities. The proposed method is an Evolutionary Algorithm-based heuristic. Computational testes were run considering a set of randomly generated instances and the results of several heuristics methods were compared with our results. The new method showed efficient in terms of solution quality and computational time.

Key-Words: Integer optimization, Cutting and packing problem, Evolutionary Algorithms.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

A.E. – Algoritmo Evolutivo

HRE – Heurística de repetição exaustiva

HRE1 - Heurística de repetição exaustiva 1

HRE2 – Heurística de repetição exaustiva 2

HRE3 – Heurística de repetição exaustiva 3

HRE4 – Heurística de repetição exaustiva 4

IIPP - *Identical Iten Packing Problem*

MBSBPP – *Multiple Bin Size Bin Packing Problem*

MHKP – *Multiple Heterogeneous Knapsack Problem*

MHLOPP – *Multiple Heterogeneous Large Object Placement Problem*

MIKP – *Multiple Identical Knapsack Problem*

MILOPP – *Multiple Identical Large Object Placement Problem*

MSSCSP – *Multiple Stock Size Cutting Stock Problem*

ODP - *Open Dimension Problem*

RBPP – *Residual Packing Problem*

RCSP – *Residual Cutting Stock Problem*

SBSBPP – *Single Bin Size Bin Packing Problem*

SKP – *Single Knapsack Problem*

SLOPP – *Single Large Object Placement Problem*

SSSCSP – *Single Stock Size Cutting Stock Problem*

SUMÁRIO

1 INTRODUÇÃO

O problema de corte, genericamente, consiste em cortar unidades maiores (objetos) em unidades menores (itens) otimizando um certo objetivo (Morabito, 1994). Como exemplos de objetivo podemos ter: a minimização da perda, a minimização do número de unidades de estoque necessárias para produzir unidades menores, ou arranjar geometricamente as unidades menores dentro das unidades maiores.

As primeiras publicações sobre o problema de corte de estoque aparecem na literatura por volta de 1940 (Kantorovich (1939) e Brooks et al (1940) apud Dyckoff e Finke (1992)), tendo ganhado mais ênfase apenas nos anos 60 com os trabalhos de Gilmore e Gomory (1961, 1963, 1965, 1966), que são modelos eficazes utilizados até os dias de hoje.

O problema aparece em diversos processos industriais em que os objetos disponíveis em estoque possuem dimensões padronizadas, como barra de aço, bobinas de papel e alumínio, placas metálicas e de madeira (Dickhoff, 1990), ou por placas de circuito impresso, chapas de vidro e fibra de vidro, corte de tubos para equipamentos de calefação (Heicken e König, 1980, apud Dickhoff, 1990), entre outros, e os itens com tamanhos especificados pelos clientes são encomendados por meio de uma carteira de pedidos (Boleta et al, 2005).

Outra abordagem pode ser vista em indústrias que possuem seus processos de produção atrelados ao empacotamento de unidades menores (itens) dentro de unidades maiores (objetos). O processo de cortar objetos em itens ou empacotar itens em objetos consiste em problemas idênticos, matematicamente falando, pois apresentam essencialmente a mesma estrutura lógica, podendo ser abordados com as mesmas formulações e estratégias de solução. Sendo assim conhecidos na literatura como Problemas de Corte e Empacotamento (PCE).

Algumas das aplicações de empacotamento podem ser vistas em: alocação de comerciais em TV, alocação de programas em discos e fitas magnéticas, carregamento de veículos, problema de programação de veículos, corte de retalhos em fábrica de tecidos, ou em confecção de roupas, empacotamento de caixas em galpões, empacotamento em contêineres (Gehring et al, 1990; Haessler e Talbot, 1990; apud Dickhoff, 1990), carregamento de cargas em furgões e corte de espumas para colchões.

Em geral, estes problemas são formulados como problemas de otimização linear inteira e são difíceis de serem tratados devido ao grande número de variáveis envolvidas e devido à restrição de integralidade das variáveis (Boleta et al, 2005), sendo problemas pertencentes à classe de problemas NP-árduo e, portanto, obter suas soluções exatas em tempos computacionais razoáveis não é uma tarefa fácil. Desta forma, torna-se interessante o uso de heurísticas para sua solução.

Segundo Boleta et al (2005) "a importância econômica aliada à dificuldade de resolução de problemas de corte e empacotamento tem motivado grande empenho da comunidade acadêmica na busca de métodos eficientes, o que pode ser notado pelo volume de publicações nos últimos anos".

Nas últimas duas décadas vários artigos, de revisão bibliográfica, relacionados ao problema de corte e empacotamento foram publicados: Dyckhoff (1990), Sweeney e Partenoster (1992), Downsland e Downsland (1992), Dyckhoff et al (1997) e Wäscher et al (2007). Alguns livros específicos ao tema podem ser citados: Martello e Toth (1990) e Dyckhoff e Finke (1992). Edições especiais de jornais, por exemplo, o *Euopean Journal of Operacional Research*, foram publicadas: Dyckhoff e Wäscher (1990), Bischoff e Wäscher (1995), Wang e Wäscher (2002) e Oliveira e Wäscher (2007).

O presente trabalho é classificado, segundo a tipologia de Wäscher et al (2007), como sendo um problema de corte de estoque de objetos de tamanhos diferentes, apresenta uma heurística alternativa as heurísticas que têm sido apresentadas na literatura, sendo uma extensão do trabalho apresentado por Boleta et at (2005). Para comparar a qualidade da solução e o tempo computacional desta nova heurística utilizaremos os resultados obtidos na literatura.

O texto está estruturado da seguinte maneira: no capítulo 2 são descritas duas classificações para os problemas de corte de estoque (Dychoff (1990) e Wäscher et al (2007)), a caracterização dos problemas de corte de estoque e uma revisão bibliográfica na área de corte é descrita. A metodologia e o algoritmo evolutivo desenvolvido são apresentados no capítulo 3. No capítulo 4 analisamos os resultados dos testes computacionais obtidos pela implementação do algoritmo evolutivo. No capítulo 5 são feitas as considerações finais, nas quais apresentamos a conclusão do trabalho juntamente com algumas propostas futuras. No Apêndice A são descritas as técnicas evolutivas utilizadas no desenvolvimento do algoritmo evolutivo proposto.

2 CLASSIFICAÇÃO, CARACTERIZAÇÃO DO PROBLEMA E REVISÃO BIBLIOGRÁFICA

Este capítulo apresenta duas classificações para o problema de corte de estoque apresentada por Dyckhoff (1990) e Wäscher et al (2007), relaciona algumas limitações da classificação proposta por Dyckhoff (1990), define o problema de corte unidimensional inteiro com objetos de tamanho diferentes, apresenta a modelagem matemática para este problema e descreve problemas correlatos na área.

2.1 Classificação dos problemas de corte

Na literatura podemos encontrar algumas formas de classificação dos problemas de corte como, por exemplo, a descrita por Dyckhoff (1990), e a de Wäscher et al (2007).

2.1.1 Tipologia de Dyckhoff (1990)

Em 1990, Dyckhoff propôs uma classificação tomando por base a estrutura lógica dos Problemas de Corte e Empacotamento e suas principais características: dimensionalidade, mensuração das quantidades, forma e sortimento dos objetos e itens, disponibilidade, restrições dos padrões, restrições de alocação e sua viabilidade. Estas características foram agrupadas em quatro critérios básicos, usados para classificar os problemas, conforme a seguir:

1. Dimensionalidade
    - (1) Unidimensional;
    - (2) Bidimensional;
    - (3) Tridimensional;
    - (*n*) *n*-dimensional.
2. Tipo de alocação
    - (B) Todos os objetos e uma parte dos itens;
    - (V) Uma parte dos Objetos e Todos os itens.

3. Sortimento dos Objetos

(O) Um objeto;

(I) Objetos de figuras idênticas;

(D) Objetos de diferentes figuras.

4. Sortimento dos itens

(F) Poucos itens (de figuras diferentes);

(M) Muitos itens de muitas figuras diferentes;

(R) Muitos itens com relativamente poucas figuras diferentes;

(C) Figuras congruentes.

Com esses critérios Dyckhoff classificou de forma consistente e sistemática os diversos tipos de Problemas de Corte e Empacotamento agrupando-os em classes, definidas por meio de uma quádrupla (a / b / g / d), onde a, b, g e d correspondem, respectivamente, aos critérios 1, 2, 3 e 4 definidos anteriormente.

Limitações da tipologia de Dyckhoff

A tipologia proposta por Dyckhoff não obteve uma unanimidade internacional. Porém, deve-se destacar que a mesma é um marco na pesquisa dos problemas de corte e empacotamento, destacando a estrutura básica comum entre os problemas de corte e os problemas de empacotamento. Algumas desvantagens e controvérsias foram apontadas por diversos pesquisadores (Wäscher et al, 2007) tais como:

- A classificação pode não ser necessariamente única. O problema do carregamento de veículos foi codificada por Dychoff (1990) com o 1/V/I/F e 1/V/I/M onde "F" caracteriza a situação em que poucos itens de diferentes formas são designados. "M" caracteriza a situação em que muitos itens de muitas diferentes formas são designados;
- A tipologia é parcialmente inconsistente, podendo apresentar resultados confusos. No caso do problema do empacotamento por faixas (*strip-packing problem*), um caso particular do problema de empacotamento bi-dimensional, onde um conjunto de itens de diferentes tamanhos é empacotado em retângulos individuais de largura fixa e comprimento mínimo, este problema foi codificado (classificado) por pesquisadores e profissionais da área como 2/V/O/M. No entanto, na tipologia de Dyckhoff ele recebe a notação 2/V/D/M. Dyckhoff (1990) justifica sua notação dizendo que este problema "....é

equivalente ao problema da seleção de variedades, onde o estoque é dado por uma infinidade de objetos de mesma largura e com todas as possibilidades de comprimento e somente um objeto será escolhido do estoque, isto é, o de menor comprimento" (Dyckhoff 1990, p.155). Dyckhoff (1990) denomina, ainda, o problema como um problema 2-D de empacotamento de bins. Denominação esta que, no entender de diversos pesquisadores (Lodi et al, 2002; Miyazawa e Wakabayashi, 2003; Faroe et al, 2003; apud Wäscher et al, 2007) seria mais óbvia reservar para a extensão natural do problema de empacotamento de *bins* unidimensional (*Bin Packing* clássico);

- A aplicação da tipologia de Dyckhoff (1990) não resulta, necessariamente, em categorias de problemas homogêneos;
- No caso do corte unidimensional, onde um grande número de itens com poucas formas diferentes são produzidos de uma quantidade ilimitada de objetos, podemos incorrer em duas situações diferentes se os objetos possuírem diferentes tamanhos: na primeira, os objetos são separados em pequenos grupos de elementos idênticos; na segunda, todos os objetos são diferentes. Ambas as situações são classificadas por Dyckhoff como 1/V/D/R. Gradisar et al (2002) consideram que utilizar a mesma classificação para as duas situações não é adequado, pois elas requerem métodos de solução diferentes (Wäscher et al, 2007).

2.1.2 Tipologia de Wäscher et al (2007)

Os problemas de corte e empacotamento, geralmente, podem ser definidos em 2 tipos, elementares ou combinados, Wäscher et al (2007). Wäscher et al (2007) utilizam esses dois tipos para desenvolvimento de modelos, algoritmos, geradores de problemas e para a categorização literária.

As definições dos tipos dos problemas são baseadas em 5 critérios, dimensionalidade, tipo de designação, variedade (sortimento) dos itens, variedade dos objetos e forma dos itens, descritos a seguir:

- dimensionalidade

  Tal como na tipoloiga de Dychoff (1990), são consideradas 1, 2 ou 3 dimensões, eventualmente são consideradas dimensões superiores (Lins et al,

2002 apud Wäscher et al, 2007). Problemas com mais de três dimensões são tratados como variações dos demais.

- tipo de designação

  Neste critério, que toma por base o critério análogo de Dyckhoff (1990), os autores se referem aos problemas como:

  - minimização do consumo (*input minization*)

    Um dado conjunto de itens deve ser designado para um conjunto de objetos. Aqui o conjunto de objetos é suficiente para acomodar todos os itens, ou seja, todos os itens serão designados para uma seleção de objetos de "valor mínimo", não existindo o problema de seleção com respeito aos itens;

  - maximização da produção (*output maximization*)

    Um dado conjunto de itens deve ser designado para um conjunto de objetos. O conjunto de objetos não é suficiente para acomodar todos os itens. Com isso, todos os objetos serão usados, para que a produção de uma seleção de itens seja maximizada.

- variedade (sortimento) dos itens

  Em relação à variedade dos itens, há uma distinção em três casos:

  - itens idênticos

    Todos os itens são da mesma forma e tamanho. Esta categoria de problemas é idêntica ao tipo C na classificação de Dyckhoff;

  - variedade fracamente heterogêneo

    Os itens podem ser agrupados em classes de itens idênticos. Por definição, itens de mesma forma e tamanho que possuem orientações diferentes são tratados como itens de tipos diferentes. A demanda de cada item é relativamente grande, podendo ser ou não limitada. Esta categoria corresponde ao tipo R na classificação de Dyckhoff;

  - variedade fortemente heterogêneo

    O conjunto de itens é caracterizado pelo fato de que poucos itens são idênticos em relação à forma e tamanho. Se isto ocorre os itens são tratados como elementos individuais e, conseqüentemente, suas demandas serão iguais a um. Esta categoria corresponde ao tipo M e F na classificação de Dyckhoff.

- Variedade de objetos

  - um objeto

    Neste caso o conjunto de objetos consiste de apenas um elemento, que poderá ter todas as dimensões fixas, ou ter uma ou mais dimensões variáveis. O primeiro caso é idêntico ao tipo O na classificação de Dyckhoff enquanto que o segundo representa uma extensão conjunto de tipos elementares de Dyckhoff;

  - vários objetos

    Neste caso não é necessário distinguir entre dimensões fixas e variáveis, com respeito aos problemas encontrados na literatura. Somente serão considerados objetos de dimensões fixas. De forma análoga às categorias introduzidas para o sortimento de itens, os objetos serão distinguidos entre idênticos, fracamente e fortemente heterogêneos. Com isso, é feita uma extensão da tipologia de Dyckhoff, que somente identifica os objetos em formas idênticas (tipo I) e formas diferentes (tipo D).

Na tipologia proposta por Wäscher et al (2007) os problemas de corte e empacotamento foram classificados em três tipos: básico, intermediário e refinado, descritos a seguir.

2.1.2.1 Tipo básico

O tipo básico leva em consideração a atribuição dos objetos e a variedade dos itens. Sendo subdivididos em duas categorias: maximização da produção e minimização do consumo:

- maximização da produção (*output maximisation*)

  O estoque dos objetos nos problemas deste tipo é limitado, conseqüentemente não é possível produzir todos os itens. Como o objetivo é maximizar a produção dos itens, segundo uma dada função objetivo, serão utilizados todos os objetos. Assim, em geral um problema de seleção dos itens é gerado, sendo que os seguintes problemas são considerados do tipo básico:

- Problema do empacotamento de itens idênticos (*IIPP – Identical Item Packing Problem*)

Esta categoria de problemas consiste na desginação do maior número possível de itens idênticos para um dado conjunto (limitado) de objetos. Como os itens são idênticos, não há um problema de seleção de itens, tão pouco há um problema de agrupamento ou de distribuição. O problema é reduzido a um arranjo dos itens (idênticos) em cada um dos objetos.

- Problema de alocação (*PP – Placement Problem*)

Uma atribuição pouco heterogênea de itens precisa ser designada a um dado conjunto (limitado) de objetos. A produção dos itens tem que ser maximizada ou, alternativamente, o desperdício correspondente minimizado.

- Problema da mochila (*KP– Knapsack Problem*)

A atribuição dos itens é muito heterogênea, que precisa ser desigando a um dado conjunto (limitado) de objetos. Busca-se maximizar os itens produzidos, conforme a função objetivo adotada.

- minimização do consumo (*input minimisation)*

Os problemas deste tipo são caracterizados pelo fato da quantidade de objetos ser grande o bastante para acomodar todos os itens. Neste caso, busca-se minimizar o valor dos objetos necessários para o atendimento à demanda, conforme a função objetivo adotada.

- Problema com dimensão variável (ODP – *Open Dimension Problem*)

Neste caso, ao menos uma das dimensões dos objetos é variável.

- Problema do corte de estoque (CSP – *Cutting Stock Problem*)

Nesta categoria uma atribuição pouco heterogênea de itens deve ser completamente alocado em uma seleção de objetos, a menor possível.

- Problema do empacotamento de *bins* (*BPP– Bin Packing Problem*)

Os problemas deste tipo possuem atribuições muito heterogêneas de itens, que devem ser alocados em um conjunto de objetos, onde o valor (número ou tamanho total) dos objetos necessários tem que ser minimizado.

A Figura 1 mostra os tipos básicos de problemas de corte e empacotamento.

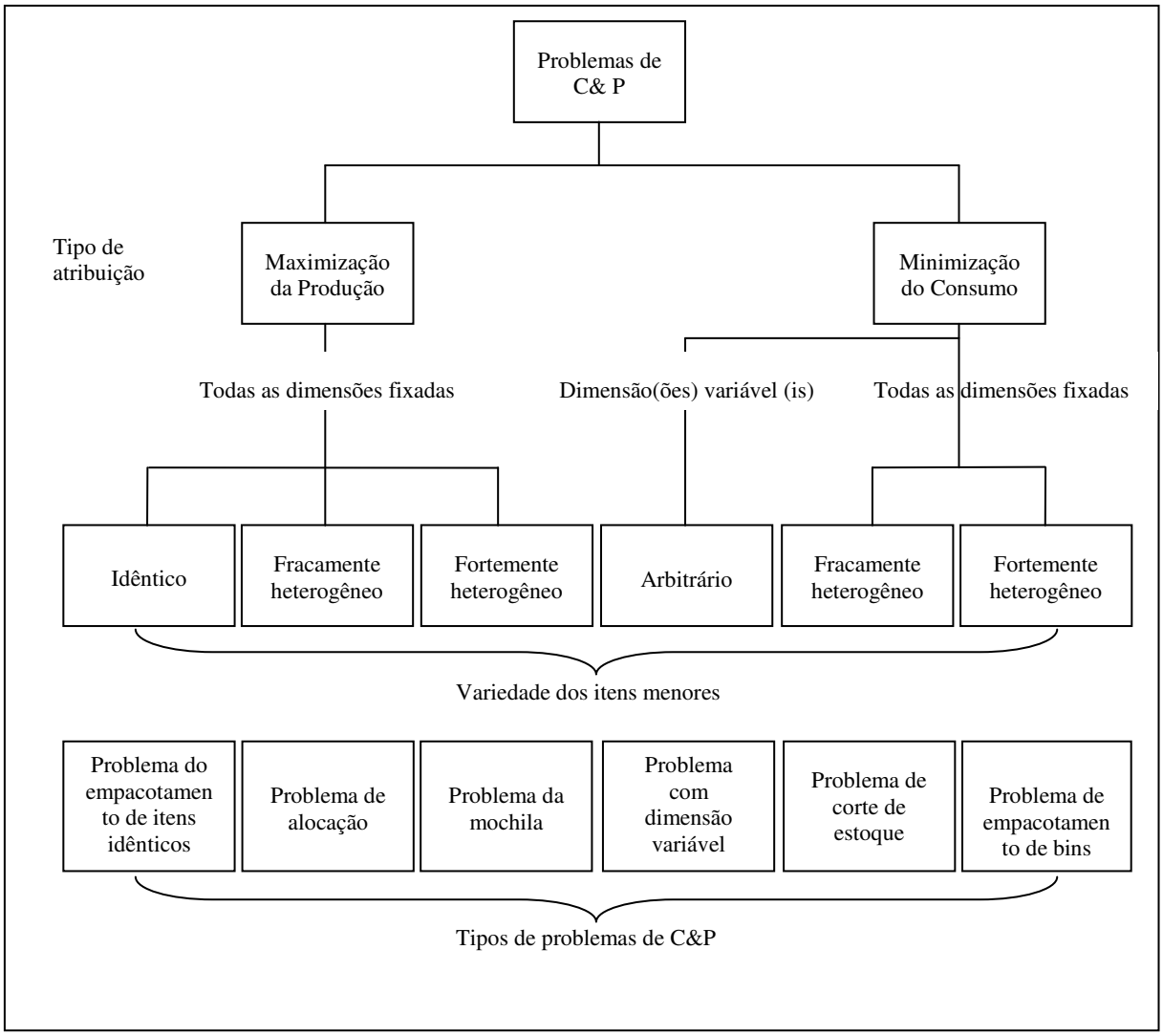


Figura 1- Tipos de problemas básicos propostos por Wäschet et al (2007).

2.1.2.2 Tipo intermediário

Para definir tipos de problemas mais homogêneos, foi adicionado um novo critério aos tipos básicos: a atribuição de objetos. Sendo assim, os tipos básicos foram reestruturados gerando uma quantidade maior de tipos.

A Tabela 1 mostra a classificação dos problemas de corte e empacotamento em relação à maximização da produção.

Tabela 1 - Tabela de tipos de problemas intermediários: maximização da produção.

Características dos objetos grandes \ Variedade dos itens menores		Idênticos	Fracamente heterogêneo	Fortemente heterogêneo
Todas as dimensões fixadas	Apenas um objeto	Problema do empacotamento de itens idênticos IIPP	Problema de alocação em um único objeto SLOPP	Problema da mochila única SKP
	Idênticos		Problema de alocação em múltiplos objetos idênticos MILOPP	Problema de múltiplas mochilas idênticas MIKP
	Heterogêneos		Problema de alocação em múltiplos objetos heterogêneos MHLOPP	Problema das múltiplas mochilas heterogêneas MHKP

A Tabela 2 mostra a classificação dos problemas de corte e empacotamento em relação à minimização do consumo.

Tabela 2- Tabela de tipos de problemas intermediários: minimização do consumo.

Características dos objetos grandes \ Variedade dos itens menores		Fracamente heterogêneo	Fortemente heterogêneo
Todas as dimensões fixadas	Idênticos	Problema do corte de estoque de objetos de tamanho único SSSCSP	Problema de empacotamento de bins de tamanho único SBSBPP
	Fracamente heterogêneos	Problema do corte de estoque de objetos de tamanhos diferentes MSSCSP	Problema de empacotamento de bins de tamanhos diferentes MBSBPP
	Fortemente heterogêneos	Problema do corte de estoque residual RCSP	Problema de empacotamento de bins residual RBPP
Apenas um objeto com dimensão variável	Problema com dimensão variável ODP		

As siglas utilizadas nas Tabelas 1 e 2 são abreviaturas do termo em inglês de cada tipo de problema. Maiores informações podem ser obtidas no trabalho original de Wäscher et al 2007.

2.1.2.3 Tipo refinado

Consiste da aplicação dos critérios de dimensionalidade e de forma dos itens, resultando em subcategorias dos problemas do tipo intermediário, que são identificadas pelos adjetivos adicionados aos seus nomes. Portanto, a estrutura do nome de um problema do tipo refinado será:

$$\{\text{dimensão}\} + \{\text{forma}\} + \{\text{IPT}\}$$

Onde:

- dimensão → 1, 2, ou 3-dimensional;
- forma → Ø, retangular, circular, cilíndrica, ..., irregular;
- IPT → nome do problema do tipo intermediário.

A tipologia proposta de Wäscher et al (2007) está baseada na tipologia de Dyckhoff (1990) introduzindo novos critérios de caracterização dos problemas, que definem as categorias dos problemas de forma diferente das apresentadas por Dyckhoff (1990), sendo mais abrangente e precisa do que a tipologia de Dyckhoff (1990).

O presente trabalho é classificado, segundo a tipologia de Wäscher et al (2007), como sendo um problema do corte de estoque de objetos de tamanhos diferentes (1D MSSCSP).

2.2 Definição do problema e modelagem matemática

Uma definição formal para o problema de corte de estoque unidimensional tratado nesta dissertação pode ser apresentada da seguinte forma (Poldi e Arenales, 2005):

> Considere que temos disponível em estoque *K* tipos de objetos (barras, bobinas, etc.) de comprimento L_k, $k = 1,...,K$, e um conjunto de *m* itens com demanda $d_i, i = 1,...,m$, e comprimentos $l_i, i = 1,...,m$ (os comprimentos dos itens são tais que: $l_i \leq L_k, k = 1,...,K, i = 1,...,m$). O problema consiste em produzir os itens demandados a partir do corte dos objetos em estoque, atendendo a demanda e minimizando a perda de material.

Neste trabalho consideramos o problema de corte de estoque limitado que ocorre quando existe restrição de estoque, ou seja, cada objeto *k* está disponível numa quantidade limitada $e_k, k = 1,...,K$. Uma segunda variante do problema, também considerada, ocorre

quando não é permitido excesso de produção, ou seja, o número total de itens cortados deve ser exatamente igual à demanda original.

Algumas definições são necessárias para dar suporte à formulação matemática do problema de corte de estoque unidimensional inteiro.

Pinto (1999) destaca que conforme o tamanho do item podemos cortar os objetos em estoque de várias maneiras diferentes. Cada maneira de cortar recebe o nome de *padrão de corte*. A Figura 2 a seguir ilustra um exemplo para o caso unidimensional para um objeto de comprimento L e itens de tamanho l_i:

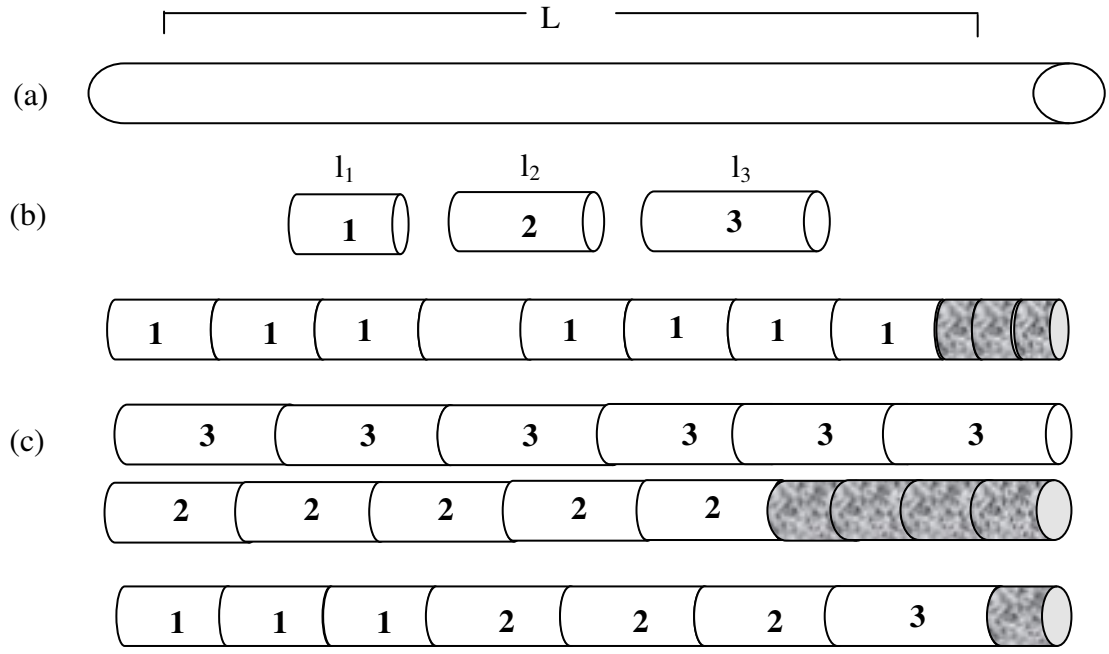


Figura 2 – Exemplo para o caso unidimensional: (a) Objeto em estoque; (b) Tipos de itens demandados; (c) Exemplos de possíveis padrões de corte.

Definição 2.1: Chamamos de *padrão de corte* a maneira como um objeto em estoque é cortado para a produção dos itens demandados. A um padrão de corte é associado um vetor m-dimensional que contabiliza os itens produzidos:

$$a = [\alpha_1, \alpha_2, ..., \alpha_m]^T$$

onde α_i é quantidade de itens do tipo i no padrão de corte a.

Definição 2.2: Um padrão de corte que produza apenas um tipo de item é chamado padrão de corte homogêneo. Um padrão de corte é homogêneo se o vetor associado tem

apenas uma coordenada não-nula: $[0, \ ..., \ \alpha_i, \ ..., \ 0]^T$, $\alpha_i \neq 0$. Note que sempre teremos *m* padrões homogêneos, cujos vetores associados definem uma matriz diagonal. A hipótese de que $l_i \leq L$, para todo *i*, está implícita, caso contrário o problema seria infactível. Os valores α_i são determinados por $\alpha_i = \left\lfloor \frac{L}{l_i} \right\rfloor, i = 1,...,m$ (Poldi, 2003).

Os padrões de corte devem ser definidos para cada tipo de barra disponível em estoque, isto é, devem satisfazer:

$$l_1\alpha_{1k} + l_2\alpha_{2k} + ... + l_m\alpha_{mk} \leq L_k \qquad (2.1)$$

$$0 \leq \alpha_{ik} \leq d_i, i = 1,...,m \text{ e inteiro}, k = 1,...,k \qquad (2.2)$$

Após a definição dos padrões de corte, o próximo passo será determinar o número de vezes que cada padrão será utilizado para resolver o problema. Assim, a modelagem matemática de um problema de corte de estoque é feita em duas etapas:

1. Definir todos os possíveis padrões de corte (supondo N_k o número total de padrões obtidos para o objeto em estoque *k, k=1, ..., K*);
2. Definir quantas vezes cada padrão de corte será utilizado (freqüência) para atender a demanda, que deverá ser um número inteiro e não-negativo.

Sejam:

$$a_{1k} = \begin{bmatrix} \alpha_{11k} \\ \alpha_{21k} \\ \vdots \\ \alpha_{m1k} \end{bmatrix}, \ a_{2k} = \begin{bmatrix} \alpha_{12k} \\ \alpha_{22k} \\ \vdots \\ \alpha_{m2k} \end{bmatrix}, \ \cdots, a_{N_k k} = \begin{bmatrix} \alpha_{1N_k k} \\ \alpha_{2N_k k} \\ \vdots \\ \alpha_{mN_k k} \end{bmatrix}, \ k = 1,..., K$$

onde α_{ijk} é o número de itens do tipo *i* no padrão de corte *j* para o objeto em estoque, *k, i=1, ...m, j=1, ...,N_k, k=1, ..., K,* (a_{jk} é o vetor correspondente ao *j-ésimo* padrão de corte sobre o objeto *k*).

Considerando um dado adicional para descrição do modelo matemático: c_{jk} que é o custo de cortar o objeto *k*, segundo o padrão de corte *j, j=1,...,*N_k*, k=1,...,K.* Por exemplo, c_{jk} é a perda no padrão de corte *j* do objeto *k*; ou c_{jk} é o número de objetos cortados, ou c_{jk} é o número de padrões diferentes de corte. A seguir o modelo matemático será apresentado.

Dados do problema:

- m: número de tipos de itens;
- l_i: comprimento do item i, $i=1, ..., m$;
- d_i: demanda do item tipo i, $i=1,..., m$;
- K: número de tipos de objetos em estoque;
- L_k: comprimento do objeto k, $k=1,..., K$;
- e_k: disponibilidade em estoque do objeto k, $k=1, ...K$;
- a_{jk} é o vetor correspondente ao *j-ésimo* padrão de corte sobre o objeto k, $j=1, ...,N_k$, $k=1, ...K$;
- c_{jk} que é o custo do objeto k, segundo o padrão de corte j, $j=1,...,N_k$, $k=1,...,K$.

Variável de decisão:

- x_{jk}: número de vezes que o objeto do tipo k é cortado usando o padrão j, $j=1, ..., N_k$, $k=1,..., K$.

O problema pode então ser formulado por:

$$\text{Minimizar } f(x_{11}, x_{12},..., x_{jk}) = \sum_{j=1}^{N_1} c_{j1}x_{j1} + \sum_{j=2}^{N_2} c_{j2}x_{j2} + ... + \sum_{j=1}^{N_K} c_{jK}x_{jK} \qquad (2.3)$$

$$\text{Sujeito a:} \quad \sum_{j=1}^{N_1} a_{j1}x_{j1} + \sum_{j=1}^{N_2} a_{j2}x_{j2} + ... + \sum_{j=1}^{N_k} a_{jK}x_{jK} = d \qquad (2.4)$$

$$\begin{aligned} \sum_{j=1}^{N_1} x_{j1} &\leq e_1 \\ \sum_{j=1}^{N_2} x_{j2} &\leq e_2 \\ \ddots &\ \vdots \\ \sum_{j=1}^{N_K} x_{jK} &\leq e_K \end{aligned} \qquad (2.5)$$

$x_{jk} \geq 0$, e inteiro $j=1,..., N_k$, $k=1,..., K$ e d vetor demanda (2.6)

Considerando as restrições do problema tem-se que a restrição (2.4) garante que a quantidade total de itens produzidos seja exatamente igual à demanda. As restrições (2.5) garantem que a quantidade de cada barra disponível em estoque não seja violada. E a

última restrição (2.6) garante que a repetição de cada padrão de corte *j* seja um número inteiro não-negativo.

Em problemas práticos, *m* (quantidade de tipos de itens) pode ser da ordem de dezenas ou centenas, enquanto que o número de possíveis padrões de corte diferentes (que é o número de colunas) pode ser da ordem de centenas de milhares, o que dificulta a resolução direta do problema.

Diferentes funções objetivo tem sido considerada na literatura, como, por exemplo,a minimização: da perda, do número de objetos cortados, do número de padrões de corte, entre outros. Além disso combinações destas funções objetivo também são consideradas.

A minimização da perda está relacionada com o desperdício de todos os objetos utilizados com os seus referidos padrões de corte, sendo o custo c_{jk} definidos de acordo com (2.7).

$$c_{jk} = \left(L_k - \sum_{i=1}^{m} l_i \alpha_{ijk} \right), k = 1,...,K \text{ e } j = 1,...,N_k \qquad (2.7)$$

Para minimizar o número de padrões de corte o custo:

$$c_{jk} = \delta(x_{jk}), k = 1,...,K \text{ e } j = 1,...,N_k \quad onde\ \delta(x_{jk}) \begin{cases} 1\ se\ x_{jk} \geq 0 \\ 0\ caso\ contrário \end{cases} \qquad (2.8)$$

2.2.1 Exemplo

O exemplo a seguir considera dois tipos de barras em estoque, isto é, barras com diferentes tamanhos (comprimentos) e três tipos de itens de tamanhos diferentes. Como dados da demanda a serem utilizados temos: número de tipos de itens será igual a 3 (*m=3*), o comprimento de cada item (l_i) e a demanda de cada (d_i) são visualizados na Figura 3.

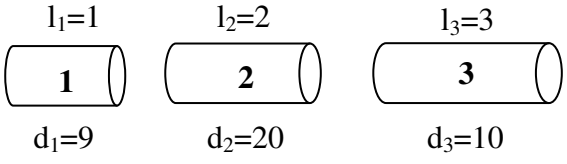


Figura 3 - Itens a serem cortados com suas respectivas demanda.

O número de tipos de barras (considerados a partir de agora como objetos) em estoque a serem cortados é igual a dois ($K=2$) e o comprimento (L_1 e L_2) e a disponibilidade (e_1 e e_2) em estoque de cada objeto é dado na Figura 4.

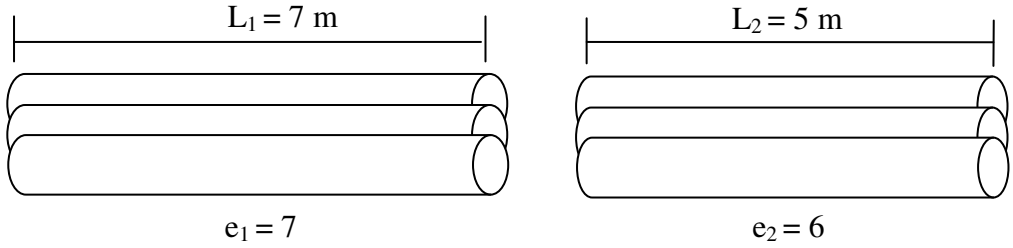


Figura 4 - Objetos em estoque a serem cortados com seus respectivos tamanhos (L_i) e quantidades disponíveis (e_i).

Cada objeto k disponível em estoque para corte poderá ser cortado de muitas formas (padrões de corte), respeitando o comprimento máximo do mesmo (L_k), atendendo a demanda solicitada com o menor desperdício possível. A Figura 5 mostra um padrão de corte para o ojbeto 1 ($k=1$). Observa-se que neste padrão de corte, padrão 1 para o objeto 1, (a_{11}), não ocorreu desperdício do objeto a ser cortado, o item 1 não foi cortado nenhuma vez, o item 2 foi cortado duas vezes e o item 3 apenas uma vez.

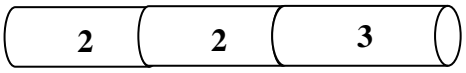


Figura 5 – Padrão de corte 1 para o objeto 1 (a_{11}).

Um segundo padrão de corte para o objeto 1 ($k=1$) pode ser obtido. O item 1 é cortado uma vez, o item 2 não é cortado, e o item 3 é cortado duas vezes. O desperdício deste objeto é nulo, isto é, não houve desperdício de material. O padrão de corte 2 para o objeto 1, (a_{21}) é representado na Figura 6.

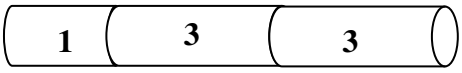


Figura 6 - Padrão de corte 2 para o objeto 1 (a_{21}).

O padrão de corte da Figura 7 pode ser obtido, utilizando o objeto 2 ($k=2$). Sendo o padrão de corte 1 para o objeto 2, (a_{12}), definido como: o item 1 é cortado uma vez, o item 2 é cortado duas vezes, o item 3 não é cortado e o desperdício da barra é zero.

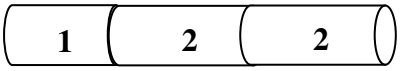


Figura 7 - Padrão de corte 1 para o objeto 2 (a_{12}).

Cada padrão de corte obtido pode ser representado por meio de um vetor. O padrão de corte 1 para o objeto 1, (a_{11}), pode ser visto em (2.9), o padrão de corte 2 para o objeto 2, (a_{21}), em (2.10) e o padrão de corte 1 para o objeto 2, (a_{12}), em (2.11).

$$a_{11} = [0 \quad 2 \quad 1]^T \qquad (2.9)$$

$$a_{21} = [1 \quad 0 \quad 2]^T \qquad (2.10)$$

$$a_{12} = [1 \quad 2 \quad 0]^T \qquad (2.11)$$

Após a definição dos padrões de corte, o próximo passo será determinar o número de vezes, (x_{jk}), que cada padrão, (a_{jk}), será utilizado para resolver o problema. Assim considerando a restrição de demanda (2.12) e de estoque (2.13) temos:

$$\begin{aligned} a_{11}x_{11} &+ a_{21}x_{21} + a_{12}x_{12} = d \\ \begin{bmatrix}0\\2\\2\end{bmatrix} x_{11} &+ \begin{bmatrix}1\\0\\2\end{bmatrix} x_{21} + \begin{bmatrix}1\\2\\0\end{bmatrix} x_{12} = \begin{bmatrix}9\\20\\10\end{bmatrix} \\ \begin{bmatrix}0\\2\\2\end{bmatrix} 4 &+ \begin{bmatrix}1\\0\\2\end{bmatrix} 3 + \begin{bmatrix}1\\2\\0\end{bmatrix} 6 = \begin{bmatrix}9\\20\\10\end{bmatrix} \end{aligned} \qquad (2.12)$$

$$\begin{aligned} x_{11} + x_{21} \le e_1 \Rightarrow 4 + 3 \le 7 \\ x_{12} \le e_2 \Rightarrow 6 \le 6 \end{aligned} \qquad (2.13)$$

2.3 Problemas Correlatos

Nesta seção serão apresentados alguns problemas correlatos na área de problema de corte de estoque, dentre eles podemos citar:

- o problema de estoque com um único tipo de barra em estoque;
- o problema de estoque com aproveitamento de sobras;
- o problema de corte de estoque unidimensional multiperíodo;
- o problema de estoque com demanda aleatória.

2.3.1 Problema de estoque com um único tipo de barra em estoque

Este tipo de problema é comum de ser encontrado na literatura: Stadtler (1990), Wäscher e Gau (1996), Poldi (2003), entre outros. O problema considera que se tem disponível um número suficientemente grande de objetos (barras, bobinas, etc.) idênticos de comprimento *L* e um conjunto de pedidos, com demanda conhecida, $d_i, i = 1,...,m$ de itens de comprimentos $l_i =, i = 1,...,m$ (os comprimentos dos itens são tais que: $l_i < L$). O problema consiste em produzir os itens demandados a partir do corte dos objetos em estoque, atendendo a demanda.

O modelo matemático para o problema de corte de estoque com um único tipo de barra é dado a seguir:

Dados do problema:

- *L*: comprimento do objeto;
- *N*: número de padrões de corte utilizados;
- a_j é o vetor correspondente ao *j-ésimo* padrão de corte sobre o objeto;
- c_j que é o custo do objeto, segundo o padrão de corte *j, j=1,...,N*.

Variável de decisão:

- x_j: número de vezes que o objeto é cortado usando o padrão *j, j=1, ..., N.*

O problema pode então ser formulado por:

$$\text{Minimizar } f(x) = \sum_{j=1}^{n} c_j x_j \qquad (2.14)$$

$$\text{Sujeito a: } \sum_{j=1}^{N_1} a_j x_j = d \qquad (2.15)$$

$x_j \geq 0$, e inteiro *j=1,..., N* e *d* vetor demanda. (2.16)

Alguns trabalhos consideram a restrição de maior e menor $(\geq)$ no lugar da igualdade $(=)$ na restrição (2.15) permitindo assim, a produção excedente de itens.

2.3.2 Problema de estoque com aproveitamento de sobras

Cherri e Arenales (2006) propõem a resolução do problema de estoque unidimensional com o reaproveitamento de sobras dos padrões de corte, desde que as sobras sejam grandes o suficiente para serem reaproveitadas no futuro para a produção itens, por exemplo, a sobra dever ser maior ou igual ao comprimento do menor item a ser produzido. Para a resolução deste problema alterações em métodos heurísticos clássicos são feitas.

2.3.3 Problema de corte de estoque unidimensional multiperíodo

Poldi e Arenales (2006) definem este tipo de problema como:

> "O problema de corte de estoque multiperíodo consiste basicamente em resolver, a cada período em um horizonte de planejamento finito, um problema de corte de estoque, para atender a uma demanda de itens naquele período do horizonte de planejamento, porém podendo antecipar ou não a produção de itens."

As sobras dos objetos não utilizados em um período podem ser reaproveitadas em um outro período, juntamente com outros objetos a serem produzidos ou adquiridos. O modelo proposto pelos autores para este problema baseia-se em um modelo de otimização linear inteira de grande porte, cujo objetivo pondera os custo com: perda de material, estocagem de objetos e de itens. O método simplex com geração de colunas foi especializado para resolver a relaxação linear.

O modelo matemático para o problema de corte de estoque multiperíodo proposto por Poldi e Arenales (2006) é dado a seguir:

Dados do problema:

- $t = 1, ...T$: número de períodos no horizonte de planejamento;
- $k = 1, ..., K$: número de tipos (comprimento) de objetos disponíveis em estoque;
- $j = 1, ..., N_k$: N_k é o número de padrões de corte para o objeto k, $k = 1, ..., K$;
- $i = 1, ..., m$: número de tipos de itens a serem cortados;
- L_k: comprimento do objeto k, $k = 1, ..., K$;

- e_{kt}: disponibilidade em estoque do objeto k no período t, $k, k = 1, ..., K, t = 1, ..., T$ (e_t: vetor de dimensão K cujas componentes são e_{kt})
- l_i: comprimento do item $i, i = 1, ..., m$;
- d_{it}: demanda do item i no período $t, i = 1, ..., m, t = 1, ..., T$ (d_t: vetor cujas componentes são d_{it});
- cap_t: capacidade das máquinas de corte no período $t, t = 1, ..., T$.
- c_{jkt}: custo de cortar o objeto k segundo o *j-ésimo* padrão de corte no período $t, j = 1, ..., N_k, k=1, ..., K, t = 1, ..., T$;
- cr_{it}: custo de estocar o item i no período $t, i = 1, ..., m, t = 1, ..., T$;
- cs_{kt}: custo de estocar o objeto k no período $t, k = 1, ..., K, t =1, ..., T$;
- b_{jkt}: custo de utilização da máquina para cortar o padrão j, do objeto k no período $t, j = 1, ..., N_k, k = 1, ..., K, t = 1, ..., T$.
- x_{jkt}: número de objetos cortados pelo padrão de corte j do objeto k no período t;
- r_{it}: número de itens do tipo i que são antecipados para o período t (r_t: vetor cujas componentes são r_{it});
- s_{kt}: número de objetos do tipo k que sobram ao final do período t.

Formulação matemática:

$$\text{minmizar } f(x) = \sum_{t=1}^{T}\left(\sum_{j=1}^{N_1} c_{j1t}x_{j1t} + \sum_{j=1}^{N_2} c_{j2t}x_{j2t} + ... + \sum_{j=1}^{N_k} c_{jkt}x_{jkt} + \sum_{i=1}^{m} cr_{it}r_{it} + \sum_{k=1}^{K} cs_{kt}s_{kt}\right)$$

$$\text{sujeito a}: \begin{cases} \sum_{j=1}^{N_1} a_{j1}x_{j1t} + \sum_{j=1}^{N_2} a_{j2}x_{2t} + ... + \sum_{j=1}^{N_k} a_{jk}x_{jkt} + r_{t-1} - r_t = d, & t = 1,...,T \\ \sum_{k=1}^{K}\sum_{j=1}^{N_k} x_{jkt} - s_{t-1} + s_t = e_t, & t = 1,...,T \\ \sum_{k=1}^{K}\sum_{j=1}^{N_k} b_{jkt}x_{jkt} \le cap_t, & t = 1,...,T \\ x_{jkt} \ge 0, \text{e inteiro}, r_{it} \ge 0, s_{kt} \ge 0, j = 1,...,N_k, k = 1,...,K, t = 1,...,T \end{cases}$$

2.3.4 Problema de estoque com demanda aleatória

No trabalho de Alem et al (2006) é apresentado um caso particular do problema de corte de estoque unidimensional sem restrições de estoque dos objetos. A demanda dos itens a serem produzidos é aleatória. A proposta do trabalho de Alem et al (2006) é estender o modelo matemático do problema de corte de estoque, cuja demanda é determinística, para o caso no qual a demanda possa assumir várias possibilidades de ocorrências. Estas possibilidades são denominadas de cenários e estes possuem probabilidades de ocorrência pré-determinadas.

2.4 Revisão Bibliográfica

Pinto (1999) relata que o problema de corte tem sido pesquisado desde o ano de 1940, sendo que as principais publicações começaram a surgir na década de 60, com os trabalho pioneiros de Gilmore e Gomory (1961, 1963 e 1965).

Na literatura são encontrados poucos trabalhos que tratam do problema de corte de estoque unidimensional com múltiplos tipos de objetos (vários comprimentos). Em 2002, esse problema foi abordado por Belov e Scheithauer (2002) e Holthaus (2002) com objetivo de minimizar a perda. Holthaus (2002) não considera as restrições de disponibilidade dos objetos em estoques, contudo, existem vários trabalhos que tratam do problema de corte unidimensional com objetos do mesmo tamanho, que são classificados por Wäscher et al (2007) como problemas de corte de estoque de objetos de tamanho único (*Single Stock Size Cutting Stock Problem, 1D SSSCSP*).

Funções objetivo diferentes tem sido consideradas na literatura e uma das mais importantes é minimizar o número de peças residuais (*trim loss*), ou seja, minimizar a perda. Considerando está função objetivo (minimizar a perda) métodos exatos e heurísticos foram desenvolvidos. Alguns métodos exatos podem ser encontrados na literatura, tais como o método proposto por Degraeve e Peters (2000), Scheithauer e Terno (1995) e Vance (1998). E métodos heurísticos são relatados na seção 2.4.1.

Segundo Umetani et al (2003) nos últimos anos outras funções objetivo foram sendo consideradas como, por exemplo o custo associado com a mudança de padrões,

sendo considerado um problema mais difícil de ser resolvido. Alguns artigos relacionados a métodos heurísticos consideram está função objetivo como, por exemplo: Haessler (1971 e 1975), Farley e Richardson (1984), Haessler e Sweeney (1991), Foerster e Wäscher (2000), Diegel et al (2006), Vanderbeck (2000), Sweeney e Hassler (1990), Goulimis (1990), Gradisar et al (1999), e Umetani et al (2003).

A seguir é descrita uma revisão dos métodos de resolução e trabalhos relacionados ao problema de corte.

2.4.1 Métodos de resolução e trabalhos relacionados ao problema de corte de estoque

Nesta seção é revisado o método Simplex com geração de colunas e os métodos heurísticos de resolução para o problema de corte de estoque. Estes métodos serão utilizados no capítulo de resultados computacionais, com o objetivo de avaliar o método proposto nesta dissertação.

Em alguns casos a resolução do problema por um modelo de otimização linear é adequada, pois a produção de algumas peças em excesso teria um baixo impacto no custo total do problema. Assim, uma abordagem prática para resolver o problema consiste em relaxar a condição de integralidade e resolver o problema relaxado pelo Método Simplex, utilizando o processo de Geração de Colunas proposto por Gilmore e Gomory (1961). Obtém-se, então, a solução ótima para o problema relaxado que, em geral, é fracionária.

Messaoud e Espinouse (2004) relatam que: “em problemas em que a demanda é baixa, a resolução por otimização linear não é suficiente”. Nestes casos, heurísticas têm sido desenvolvidas para obter soluções inteiras.

2.4.1.1 Método Simplex

O Método Simplex publicado por George B. Dantzig, em 1947, foi o marco definitivo para a disseminação da Pesquisa Operacional. Este método consiste em um algoritmo para resolução de Problemas da Programação Linear (PPL), sendo este considerado a primeira técnica da Pesquisa Operacional (Dantzig, 1981/1982)

Revisão do método

Considerando o problema primal de otimização na forma padrão:

$$\begin{aligned} &\min f(x) = c^T x \\ &\text{sujeito a}: Ax = b \\ &\quad x \geq 0 \end{aligned} \qquad (2.17)$$

A solução geral do sistema em (2.17) pode ser descrita da seguinte forma. Considere uma partição nas colunas de *A*:

$$A = (B, N)$$

Tal que $B \in R^{mxm}$, formada por m colunas da matriz A, seja inversível. A mesma partição é feita no vetor das variáveis.

$$x = (x_B, x_n)$$

E x_B é chamado vetor de variáveis básicas e x_N vetor de variáveis não-básicas (ou variáveis livres).

Assim,

$$Ax = b \Leftrightarrow Bx_B + Nx_N = b \Leftrightarrow \; x_B = B^{-1}b - B^{-1}Nx_N \qquad (2.18)$$

A expressão (2.18) é chamada solução geral do sistema, ou seja, com ela podemos determinar qualquer solução do sistema pois, para quaisquer valores atribuídos às n-m variáveis não básicas em x_N, as m variáveis básicas de x_B ficam unicamente determinadas.

Definição 1: A solução particular de x obtida por : $x_B^0 = B^{-1}b$ e $x_n = 0$, é chamada solução básica. Se $x_B^0 = B^{-1}b \geq 0$, então a solução é básica primal-factível e dizemos que a partição básica também é primal-factível.

Considere também a partição básica nos coeficientes da função objetivo c^T :

$$c^T = (c_B, c_N)^T$$

A função objetivo $f(x) = c^T x$ pode ser expressa em função das variáveis básicas e não-básicas. Assim:

$$f(x) = c_B^T x_B + c_N^T x_N \qquad (2.19)$$

substituindo 2.18 em 2.19, temos:

$$f(x) = c_B^T (B^{-1}b - B^{-1}Nx_N) + c_N^T x_N \; = c_B^T B^{-1}b - c_B^T B^{-1}Nx_N + c_N^T x_N \qquad (2.20)$$

Note que o primeiro termo de (2.20) corresponde ao valor da função objetivo na solução básica, pois:

$$x_B^0 = B^{-1}b \text{ e } x_n = 0 \Leftrightarrow$$
$$f(x_B^0) = c_B^T x_B^0 + c_N^T x_N = c_B^T B^{-1}b + c_N^T [0]$$
$$= c_B^T B^{-1}b$$

Definição 2: Chamamos o vetor $\pi \in R^m$, dado por

$$\pi^T = c_B^T B^{-1}$$

por vetor multiplicador simplex.
Temos, então:

$$f(x) = f(x_B^0) - \pi^T Nx_N + c_N^T x_N$$
$$= f(x_B^0) + [c_N^T - \pi^T N]x_N$$

ou ainda, efetuando o produto dos vetores acima, considerando que:

$$c_N^T - \pi^T N = (c_{N1} - \pi^T a_{N1}, c_{N2} - \pi^T a_{N2}, \ldots, c_{Nn-m} - \pi^T a_{Nn-m}) \text{ e}$$

$x_N = (x_{N1}, x_{N2}, \ldots, x_{Nn-m})^T$, obtemos:

$$f(x) = f(x_B^0) + (c_{N1} - \pi^T a_{N1})x_{N1} + (c_{N2} - \pi^T a_{N2})x_{N2} + \ldots + (c_{Nn-m} - \pi^T a_{Nn-m})x_{Nn-m} \quad (2.21)$$

na qual os índices N_1, N_2, ... N_{n-m}, são índices das variáveis não-básicas. Além disso, a matriz não-básica N é dada por:

$$N = [a_{N1}, a_{N2}, \ldots, a_{Nn-m}]$$

onde a_{Nj} é *j-ésima* coluna de *N*.

A expressão (2.21) mostra como a função objetivo se altera quando são feitas mudanças nos valores da variáveis não-básicas. Os coeficientes das variáveis não básicas $x_j : c_j - \pi^T a_j$ são chamados custos reduzidos.

Como objetivo é minimizar *f(x)* e sabemos que $xj \geq 0$ para todo $j \in N$, a componente x_k deve ser tal que $c_K - \pi^T a_K < 0$. A estratégia simplex é alterar as variáveis não-básicas que valem zero para a solução básica.

Teorema: Uma condição suficiente para que uma solução básica factível seja ótima é dada por:

$$c_j - \pi^T a_j \geq 0, \ j = 1, \ldots, n-m.$$

Fazendo $x_k = \varepsilon \geq 0$ e $x_j = 0, \forall j \in N - \{k\}$, as variáveis básicas são alteradas resultando em nova solução:

$$x_B = x_B^0 + \varepsilon y$$
$$x_N = \varepsilon e_k$$

Com isso a função objetivo vale:

$$f(x) = f(x^0) + (c_{N1} - \pi^T a_{N1})0 + \ldots + (c_{Nk} - \pi^T a_{Nk})\varepsilon + \ldots + (c_{Nn-m} - \pi^T a_{Nn-m})0$$
$$= f(x^0) + (c_{Nk} - \pi^T a_{Nk})\varepsilon < f(x^0)$$

A função decresce quando ε cresce. Assim, ε deve assumir o maior valor mantendo factibilidade.

Tamanho do passo ε:

Com a mudança nas variáveis não-básicas, dada pela estratégica simplex, as variáveis básicas, x_B devem também ser alteradas, de modo que o sistema $Ax = b$ seja satisfeito.

Sendo:

$$x_n = \begin{bmatrix} x_{N1} \\ \vdots \\ x_{Nk} \\ \vdots \\ x_{Nn-m} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ \varepsilon \\ \vdots \\ 0 \end{bmatrix}$$

então:

$$x_B = B^{-1}b - B^{-1}Nx_N = x^0 - B^{-1}Na_{Nk}\varepsilon = x^0 + d\varepsilon \geq 0$$

onde:

$$y = -B^{-1}a_{Nk}$$

Reescrevendo a equação vetorial acima, em cada uma de suas coordenadas, temos:

$$x_{Bi} = x_{Bi}^0 + y\varepsilon \geq 0\,,\ i = 1,\ldots,m$$

Assim se:

$$yi \geq 0 \Rightarrow x_{Bi} \geq 0\,,\ \forall \varepsilon \geq 0$$

$$yi < 0 \Rightarrow x_{Bi} = \hat{x}_{Bi} + y\varepsilon \geq 0\,, \Rightarrow \varepsilon \leq \frac{-x_{Bi}^{0}}{y_i}$$

Portanto, o maior valor de ε é dado por:

$$\hat{\varepsilon} = \min\{\frac{-x_{Bi}^{0}}{y_i} \quad tal \quad que \quad y_i < 0\} = \frac{-x_{Bi}^{0}}{y_i}$$

com este valor de $\varepsilon = \hat{\varepsilon}$, a variável básica x_{Br} se anula e a variável não básica x_{Nk} torna-se positiva. Isto sugere uma nova partição com os índices trocados:

$$B_r \Leftrightarrow N_K$$

Tal que a matrizes básicas e não-básicas sejam alteradas em apenas uma coluna:

$$B = [a_{B1}, \ldots, a_{Br}, \ldots a_{Bm}] \rightarrow B' = [a_{B1}, \ldots, a_{Nk}, \ldots a_{Bm}]$$
$$N = [a_{N1}, \ldots, a_{Nk}, \ldots a_{Nn-m}] \rightarrow N' = [a_{N1}, \ldots, a_{Br}, \ldots a_{Nn-m}]$$

De fato, pode-se mostrar que a nova partição é básica, isto é, as colunas de B´ são linearmente independentes e, além disso, a solução obtida básica associada é aquela obtida pela estratégia simplex, a qual é uma solução factível, por construção. Com isto, mostramos que a estratégia simplex produz uma nova solução básica factível onde a função objetivo tem um valor menor.

Geração de Colunas

O procedimento consiste em gerar um coluna *k*, isto é, um padrão de corte, utilizando o critério de Dantizg, que procura a variável x_k com o menor custo relativo, o que sugere o seguinte sub-problema:

$c_k - \pi^T a_k = mín\{c_j - \pi^T a_j \geq 0, j = 1,\ 2,\ ...\}$ onde π é vetor multiplicador simplex.

Considere a=$(\alpha_1, \alpha_2, \ ..., \alpha_m)$, uma coluna de *A* e suponha que os coeficientes na função objetivo sejam dados por:

$$c(a) = \gamma_0 + \sum_{i=1}^{m} \gamma_i \alpha_i,$$

onde γ_i são conhecidos. Assim, o custo relativo da variável, cuja coluna é dada por a, pode ser determinado por:

$$\varphi(a) = c(a) - \pi^T a = \gamma_0 + \sum_{i=1}^{m} \gamma_i \alpha_i - \sum_{i=1}^{m} \pi_i \alpha_i = \gamma_0 + \sum_{i=1}^{m} (\gamma_i - \pi_i) \alpha_i$$

e o sub-problema, para determinar a coluna de A a entrar na base, pode ser escrito como:

$$\text{minimizar } \varphi(a) = \gamma_0 + \sum_{i=1}^{m} (\gamma_i - \pi_i) \alpha_i$$

$$\text{sujeito a: } (\alpha_1, \alpha_2, \ ..., \alpha_m) \in X.$$

As colunas da matriz A, não estando disponíveis, serão geradas pela resolução do sub-problema.

2.4.1.2 Heurísticas de repetição exaustiva (HRE)

As heurísticas de repetição exaustiva (Hinxman, 1980) têm como procedimento básico construir um bom padrão de corte e usar este padrão tanto quanto possível, posteriormente atualiza-se a demanda e o estoque, e o procedimento é repetido gerando-se um novo padrão de corte até que algum critério de parada seja alcançado. Estas heurísticas de repetição exaustiva são bem conhecidas na literatura: Hinxman (1980), Stadtler (1990), Wäscher e Gau (1996) e Pinto (1999), e têm a seguinte estrutura geral:

Enquanto houver demanda faça:

1. construa um bom padrão de corte;
2. use o padrão do passo 1 tanto quanto for possível, sem gerar excesso de itens cortados;
3. atualize a demando do estoque.

Fim enquanto.

Para construir um bom padrão de corte no passo 1 utilizam-se heurísticas de construção que obtém gradativamente uma solução factível, ou seja, tem-se um processo iterativo que inicia com um padrão vazio e adiciona-se um novo elemento a cada iteração até a obtenção de um padrão.

Apresentaremos a seguir duas heurísticas de construção clássicas, bem conhecidas na literatura (Wäscher e Gau (1996), Pinto (1999)), que são as heurísticas FFD (*First-Fit-Decreasing*) e gulosa. Além disso, apresentaremos um algoritmo para a heurística de repetição exaustiva, utilizando estas duas heurísticas de construção.

2.4.1.2.1 Heurística de repetição exaustiva utilizando FFD (*First-Fit Decreasing*)

A heurística FFD (*First-Fit Decreasing*) sempre encontra um padrão factível para o problema de corte de estoque unidimensional (Pinto, 1999) e Wäscher e Gau (1996)). De forma geral, a heurística FFD consiste em colocar o maior item num padrão de corte tantas vezes quanto for possível, ou seja, até que não haja mais espaço para colocar este item ou, até que sua demanda já tenha sido atendida (os itens maiores são alocados em primeiro lugar, pois são os mais difíceis de serem combinados). Quando não for mais possível colocar o item maior, o segundo maior item é considerado e assim sucessivamente. A heurística de repetição exaustiva utilizando FFD repetirá o procedimento da heurística FFD enquanto existir demanda residual a ser atendida (demanda diferente de zero).

Para ficar mais claro como a heurística de repetição exaustiva utilizando FFD (HRE1) funciona apresentamos o seu algoritmo:

Algoritmo: Heurística de Repetição Exaustiva utilizando FFD (HRE1)

Início do algoritmo

Passo 1: ordene as peças a serem demandadas, segundo o seu tamanho, em ordem decrescente.

$$l_1 \geq l_2 \geq \ldots \geq l_m$$

Faça $r_i = d_i \; i = 1,\ldots,m; s_k = e_k \; k = 1,\ldots,K; \; j = 1; D = \sum_{i=1}^{m} r_i$

Passo 2: *Enquanto* $D > 0$ *(enquanto existir demanda a ser satisfeita) faça*

Para $k = 1,...,K$

se $s_k > 0$

Faça $i = 1; RESTO = L_k$

Enquanto $i \leq m$ e $RESTO \geq l_m$ faça:

$$\alpha_{ik} = \min\left\{\left\lfloor \frac{RESTO}{l_i} \right\rfloor, r_i\right\}$$

RESTO = RESTO - $\alpha_{ik} * l_i$

i = i + 1

{fim enquanto}

{fim se}

$w_k = l_k - RESTO$

{fim para}

{fim enquanto}

Passo 3: Determine *p* tal que: $w_p = \min\{w_k, \text{ tal que } s_k > 0, k = 1,...,K\}$

Determine o número de vezes que cada padrão será utilizado:

$$x_{jp} = \min\left\{\left\lfloor \frac{r_i}{\alpha_{ip}} \right\rfloor, s_p, \text{ onde } \alpha_{ip} \neq 0\, i = 1,...,m\right\}$$

Passo 4: {atualização}

$r_i = r_i - x_{jp} * \alpha_{ip}\ i = 1,...,m$

$D = D = -\sum_{i=1}^{m} x_{jp}(\alpha_{ip})$

$s_p = s_p - x_{jp}\ i = 1,...,m$

$j = j + 1$

{fim enquanto (Passo 2)}

Fim do algoritmo.

A utilização do algoritmo acima descrito gera padrões para cada tipo de objeto disponível em estoque. Determina-se o padrão que possui a menor perda entre todos os objetos. Atualiza-se a demanda residual[1] de cada item, o estoque dos objetos e a demanda residual total. O procedimento é repetido até que a demanda residual total seja zero.

2.4.1.2.2 Heurística Gulosa

É uma heurística de repetição exaustiva (Hinxman, 1980) que sempre produz uma solução inteira e pode ser aplicada tanto ao problema residual final como ao problema original. A idéia desta heurística é gerar um bom padrão de corte e utilizá-lo à exaustão (sem que haja excessos).

O critério de otimização de um algoritmo guloso é meramente local, não sendo possível, portanto, garantir a solução ótima global. A cada iteração o algoritmo considera apenas a próxima decisão, levando a ser chamado também de algoritmo míope, pois "enxerga" somente o que está mais próximo.

A heurística gulosa para o problema de corte unidimensional inteiro, consiste em resolver o problema da mochila[2] (2.22) com uma função objetivo apropriada, o valor de utilidade $v_i = l_i$, o que equivale a minimizar a perda no padrão. Para outros valores de v_i, veja Pillegi (2002).

$$\begin{aligned} &\text{maximizar } \mathrm{z}(\mathrm{x}) = v_1 x_1 + v_2 x_2 + \ldots + v_m x_m \\ &\text{sujeito a } \begin{cases} l_1 x_1 + l_2 x_2 + \ldots l_m x_m \le L \\ 0 \le xi \le ri\,,\ i = 1, \ldots, m \ \text{ e inteiros.} \end{cases} \end{aligned} \qquad (2.22)$$

A cada iteração gerada um padrão de corte é obtido e utilizado tantas vezes quanto possível. Em seguida a demanda e o estoque são atualizadas e o procedimento é repetido, gerando um novo padrão de corte, até que toda a demanda seja satisfeita ou o estoque acabe.

---

[1] Demanda residual é a demanda atualizada do item, ou seja, a demanda inicial do item menos a demanda alocada para o item.

[2] Dada uma mochila de capacidade limitada de peso e um conjunto itens distintos com pesos conhecidos e cujo transporte resulta num determinado lucro, encontre uma combinação dos itens a serem transportados, maximizando o lucro total.

O algoritmo construtivo guloso é o mesmo da heurística de repetição exaustiva utilizando FFD, a ordenação no **Passo 1** do algoritmo descrito na seção 2.4.1.2.1 não é necessário (somente a inicialização das variáveis), a seguir é apresentado o **Passo 2** com as mudanças necessárias.

Passo 2: { Construir um bom padrão de corte para cada objeto *k* em estoque, $k = 1,...,K$ }

Para $k = 1,...,K$ faça

Se $s_k > 0$ então

Resolver o problema da mochila:

$$\text{maximizar } v_k = l_1\alpha_{1k} + l_2\alpha_{2k} + ... + l_m\alpha_{mk}$$

$$\text{sujeito a } \begin{cases} l_1\alpha_{1k} + l_2\alpha_{2k} + ... + l_m\alpha_{mk} \le L_k \\ 0 \le \alpha_{1k} \le r_i, \alpha_{ik} \text{ inteiro}, i = 1,...,m; \end{cases}$$

faça: $w_k = L_k - v_k$; {perda no padrão de corte}

fim se;

fim para;

2.4.1.3 Heurísticas residuais

As heurísticas residuais consistem em obter uma solução para o problema relaxado (2.3 a 2.6) e arredondar as soluções fracionárias para baixo (maior inteiro inferior), posteriormente, resolve-se novamente o problema relaxado considerando apenas a demanda residual (ou seja, o que não foi produzido na primeira resolução devido ao arredondamento). Este procedimento é repetido até que se tenha uma solução factível.

Em notação matricial o modelo matemático (2.3 a 2.6) é escrito como:

$$\text{minimizar } f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} \qquad (2.23)$$

$$\text{sujeito a : } \begin{cases} \mathbf{Ax} = \mathbf{d} & (2.24) \\ \mathbf{Ex} \le \mathbf{e} & (2.25) \\ \mathbf{x} \ge \mathbf{0}, \text{ inteiro} & (2.26) \end{cases}$$

onde **A** é uma matriz de padrões de corte em (2.4) e **E** é uma matriz de 0´s e 1´s de (2.5).

Definição do problema residual: seja *y* uma solução inteira aproximada de **x**, **r** = **d** – **Ay** a demanda residual e **s** = **e** – **Ey** o residual dos objetos em estoque avaliados. O problema residual é dado por:

$$\text{minimizar } f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} \qquad (2.27)$$

$$\text{sujeito a : } \begin{cases} \mathbf{Ax} = \mathbf{r} & (2.28) \\ \mathbf{Ex} \le \mathbf{s} & (2.29) \\ \mathbf{x} \ge \mathbf{0}, \text{ inteiro} & (2.30) \end{cases}$$

2.4.1.3.1 Heurísticas residuais – procedimento básico

A estrutura geral para uma heurística residual é a seguinte:

Passo 1: { *Inicialização* }

Sejam $k = 0$, $\boldsymbol{r}^0 = \boldsymbol{d}$ e $\mathbf{s^0} = \mathbf{e}$, os dados para o problema residual inicial.

Passo 2: { *Determinar uma solução ótima contínua* }

Resolver o problema (2.23)–(2.26) com $\boldsymbol{r}^k = \boldsymbol{d}$, e $\boldsymbol{s}^k = \boldsymbol{e}$ { *método Simplex com geração de colunas* }

Seja $\boldsymbol{x}^k$ a solução contínua obtida. (a técnica de geração de colunas é usada)

Se $\boldsymbol{x}^k \in Z^n$, ou seja, é uma solução inteira, então PARE.

Passo 3: { *Determinar uma solução inteira aproximada* }

Determine uma solução inteira aproximada para $\boldsymbol{x}^k$ e denote-a por $\boldsymbol{y}^k$.

Se $\boldsymbol{y}^k$ for um vetor nulo, então vá para o passo final.

Passo 4: { *Atualizações* }

Determine a nova demanda residual e estoque:

$\boldsymbol{r}^{k+1} = \boldsymbol{r}^k - \boldsymbol{A}\boldsymbol{y}^k$.

$\boldsymbol{s}^{k+1} = s^k - \boldsymbol{E}\boldsymbol{y}^k$

$k = k + 1$.

Repita o **Passo 2**.

Passo Final:

Se o procedimento finalizou no passo 2, então uma solução para o problema é obtida.

Senão, (finalizou no passo 3) restou um problema residual.

Este algoritmo será completamente definido quando especificarmos como determinar a solução inteira aproximada $\mathbf{y}^k$ no **Passo 3** e como resolver o problema residual final no **Passo Final**.

As três próximas heurísticas a serem definidas: residual FFD, residual gulosa e residual *bin-packing* têm o mesmo **Passo 3** no algoritmo geral descrito na seção anterior. A solução inteira aproximada $\boldsymbol{y}^k$ é determinada arredondando-se para o inteiro inferior a solução contínua $\boldsymbol{x}^k$. Esta solução aproximada pode levar ao problema residual final prematuramente, em caso de baixa demanda.

2.4.1.3.2 Heurística residual FFD

Passo 3: { *Determinar uma solução inteira aproximada:* $\boldsymbol{y}^k$ }

Arredondar as componentes do vetor $x^k \in \Re^n$ para o maior inteiro menor ou igual a x^k, isto é: $y^k = \lfloor x^k \rfloor$.

Passo Final: Resolver o problema residual final pela heurística de repetição exaustiva utilizando FFD, descrita na seção 2.4.1.2.1.

2.4.1.3.3 Heurística residual gulosa

Passo 3: { *Determinar uma solução inteira aproximada:* $\boldsymbol{y}^k$ }

Arredondar as componentes do vetor $x^k \in \Re^n$ para o maior inteiro menor ou igual a x^k, isto é: $y^k = \lfloor x^k \rfloor$

Passo Final: Resolver o problema residual final pela heurística construtiva gulosa (seção 2.4.1.2.2).

2.4.1.3.4 Heurística residual utilizando o empacotamento de bins (*residual bin packing*)

Passo 3: { *Determinar uma solução inteira aproximada:* $\boldsymbol{y}^k$ }

Arredondar as componentes do vetor $x^k \in \Re^n$ para o maior inteiro menor ou igual a x^k, isto é: $y^k = \lfloor x^k \rfloor$.

Passo Final: O problema de corte de estoque residual final (poucos itens restantes) é modelado como um problema *bin-packing* e resolvido de forma exata. A seguir descreveremos este procedimento.

O problema de empacotamento de *bins* (*bin packing problems*)

Os objetos em estoque restantes e os itens restantes ordenados são numerados um por um, para todo caso de repetição. Por simplicidade de notação, seja n a quantia total de objetos em estoque restante (possibilitando, $n = \sum_{k=1}^{K} s_k^l$, quando l for a última iteração do algoritmo residual) e m' a quantia total de itens restantes ordenados ($m' = \sum_{i=1}^{m} r_i^l$). O comprimento do objeto em estoque do tipo j é representado por L_j e o tamanho do item tipo i é representado por l_i.

Sejam as variáveis de decisão definidas por:

$$y_i = \begin{cases} 1, \text{ se o objeto } j \text{ é usado,} \\ 0, \text{ caso contrário} \end{cases}$$

$$x_{ij} = \begin{cases} 1, \text{ se o item } i \text{ é cortado do objeto } j, \\ 0, \text{ caso contrário} \end{cases}$$

O problema consiste em (Martello e Toth, 1990):

$$\text{minimizar} \quad g(y) = \sum_{j=1}^{n} y_j \tag{2.31}$$

$$\text{sujeito a:} \quad \sum_{i=1}^{m'} \ell_i x_{ij} \leq L\, y_j \qquad j = 1, \ldots, n \tag{2.32}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad i = 1, \ldots, m' \tag{2.33}$$

$$y_j = \{0,1\}, \;\; x_{ij} = \{0,1\} \qquad i = 1, \ldots, m', \; j = 1, \ldots, n. \tag{2.34}$$

Observe que o modelo (2.31)-(2.34) pode ser usado para modelar o problema original, porém com um alto número de itens (aqui considerados um a um, ignorando repetições). Para os problemas residuais, os quais envolvem poucos itens, o modelo (2.31)-(2.34) pode ser resolvido por um método de enumeração implícita. Entretanto, os experimentos computacionais atestam a dificuldade desta abordagem devido alta simetria dos objetos (Poldi e Aranelas, 2005).

2.4.1.4 Heurística de Poldi e Arenales (2005)

A heurística proposta por Poldi e Arenales (2005) difere das heurísticas residuais tradicionais na maneira de determinar a solução inteira aproximada, no **Passo 3** do algoritmo apresentado na seção 2.4.1.2.1, nas quais um arredondamento simples é sempre feito. A heurística é baseada na idéia de que o arredondamento para o inteiro superior é infactível para todas as freqüências, mas algumas delas poderiam ser arredondadas para o inteiro superior, enquanto outras seriam arredondadas para o inteiro inferior.

Existem três versões da nova heurística proposta por Poldi e Arenales (2005) os quais são baseadas nesta idéia. Assim as heurísticas consistem basicamente em, a cada iteração, resolver um problema de corte de estoque relaxado e ordenar o vetor solução de forma específica (Poldi e Arenales (2005) analisaram três formas de ordenação). Para cada posição deste vetor, na ordem especificada, arredonda-se a freqüência para o número inteiro acima do fracionário obtido e testa-se a factibilidade desta solução (no sentido de que excessos de itens não sejam gerados). Caso não seja factível (isto é, houve excesso de itens), a freqüência é reduzida de uma unidade até que excessos sejam eliminados. Quando o último padrão de corte gerado for examinado, atualiza-se a demanda, resultando num problema residual, que será tratado da mesma forma. É importante notar que no processo de geração de colunas, ao menos um dos padrões de corte gerados pode ser utilizado pelo menos uma vez. Isto garante que a demanda residual fica cada vez menor a cada iteração e, ao final, a demanda residual é nula. Ou seja, o passo final é desnecessário.

Para definir esta nova heurística, o **Passo 3** do algoritmo residual geral, descrito na seção 2.4.1.2.1 foi alterado. O **Passo 3** foi dividido em duas partes: o pré-processamento (ordenação dos padrões) e o arredondamento. O novo algoritmo residual é descrito a seguir:

Passo 3: { *Determinar uma solução inteira aproximada* }

Passo 3.1 *{pré-processamento}*

Ordene o vetor solução $\mathbf{x}^k \in \Re^n$ obtido no Passo 2 segundo um critério a ser definido a seguir para cada uma das três versões da heurística nova.

Passo 3.2 *{arredondamento}*

Suponha que os padrões de corte gerados na solução ótima no **Passo 2** tenham freqüência positiva $\left(x_1^k > 0,...,x_N^k > 0\right)$.

Para $i = 1,...,N$ faça

$y_{ik_i} = \min\{\lceil x_{ik_i} \rceil, s_{k_i}\}$

$s_{k_i} = s_{k_i} - y_{ik_i}$ {atualiza o estoque residual}

enquanto $\mathbf{Ay}$ não for menor ou igual a $\mathbf{r}$, faça {enquanto $\mathbf{y}$ for infactível, há excesso}

$y_{ik_i} = y_{ikI}$-1

$s_{ikI} = s_{ik_i} + 1$

Fim.

Após $i = N$ (número de padrões gerados), todas as freqüências foram 'arredondadas' (algumas freqüências foram arredondadas possivelmente para o inteiro superior, outras para o interior inferior ou valores menores, podendo-se até mesmo anular uma freqüência maior que um), gerando-se uma solução aproximada inteira $\mathbf{y}^k$.

As três novas heurísticas serão apresentadas a seguir.

2.4.1.4.1 Heurística nova 1

A estratégia da heurística residual nova 1 é apostar na qualidade dos padrões gerados pelo modelo de otimização linear, no qual toda a demanda é levada em conta na construção dos padrões de corte.

Para isso basta redefinir o **Passo 3.1** do algoritmo anterior, que trata da ordenação dos padrões de corte. Na versão 1, os padrões de corte são ordenados de forma não-decrescente de freqüência de utilização, ou seja, é dado maior prioridade aos padrões mais utilizados.

Passo 3.1: *{pré-processamento: prioridade com maior freqüência}*

Ordene o vetor solução $\mathbf{x}^k \in \Re^n$ obtido no **Passo 2** tal que: $x_{1k_1} \geq x_{2k_2} \geq ... \geq x_{Tk_T}$.

2.4.1.4.2 Heurística nova 2

Para definir a heurística residual nova 2, basta redefinir o **Passo 3.1** do algoritmo anterior, que trata da ordenação dos padrões de corte. A heurística nova 2 ordena os

padrões de corte conforme a perda que eles apresentam, de forma não-crescente.Assim, tenta-se utilizar primeiro os padrões de corte com a menor perda, caso haja empate, o critério de desempate é colocar primeiro o padrão com maior freqüência x_j.

Passo 3.1: *{pré-processamento: prioridade com custo menor}*

Seja w_j a perda no padrão de corte $j, j = 1,...,N$.

Ordene o vetor solução $\mathbf{x}^k \in \Re^n$ obtido no **Passo 2** tal que: $w_{1k_1} \leq w_{2k_2} \leq ... \leq w_{Tk_T}$.

2.4.1.4.3 Heurística nova 3

Nesta heurística a ordenação dos padrões de corte definidos no **Passo 3.1** é feita conforme a parte fracionária das freqüências dos padrões de corte.

Passo 3.1: *{pré-processamento: maior valor da parte fracionária}*

Seja $f_{1jk_1} = x_{jk} - \lfloor x_{jk} \rfloor, j = 1,...,N.$ (*f* é parte fracionária).

Ordene o vetor solução $\mathbf{x}^k \in \Re^n$ obtido no **Passo 2** tal que $f_{1k_1} \geq f_{2k_2} \geq ... \geq f_{Tk_T}$.

3 UM ALGORITMO EVOLUTIVO PARA O PROBLEMA DE CORTE DE ESTOQUE UNIDIMENSIONAL INTEIRO

A heurística aqui descrita é uma extensão do trabalho proposto por Boleta et al (2005), o qual é baseado em algumas características dos Algoritmos Genéticos e consiste, primeiramente, em gerar uma população inicial com indivíduos bons. Para isso foram utilizados métodos clássicos da literatura, os quais demandam um tempo computacional grande para a geração da população inicial. Posteriormente, tem-se a aplicação de mecanismos evolutivos para gerar novos indivíduos. Boleta et al (2005) destacam que: "a vantagem resultante deste método é a qualidade da solução (embora tenham sido feitos poucos testes, a heurística proposta mostrou-se capaz de melhorar bastante os resultados de métodos simples) e como desvantagem tem-se os tempos computacionais".

3.1 Metodologia e desenvolvimento do trabalho

A Figura 8 mostra a visão geral da metodologia aqui proposta. No passo 1 é feita a geração inicial da população de indivíduos, que são soluções factíveis. A seguir é apresentado critério de parada do algoritmo evolutivo proposto. O passo 2 mostra o processo de evolução utilizado levando em conta: a seleção de indivíduos, a cooperação entre os indivíduos para gerar um novo indivíduo e o processo de adaptação dos indivíduos. Se a nova solução obtida for melhor do que a pior solução da população a mesma será substituída, caso contrário o processo de reprodução é executado novamente.

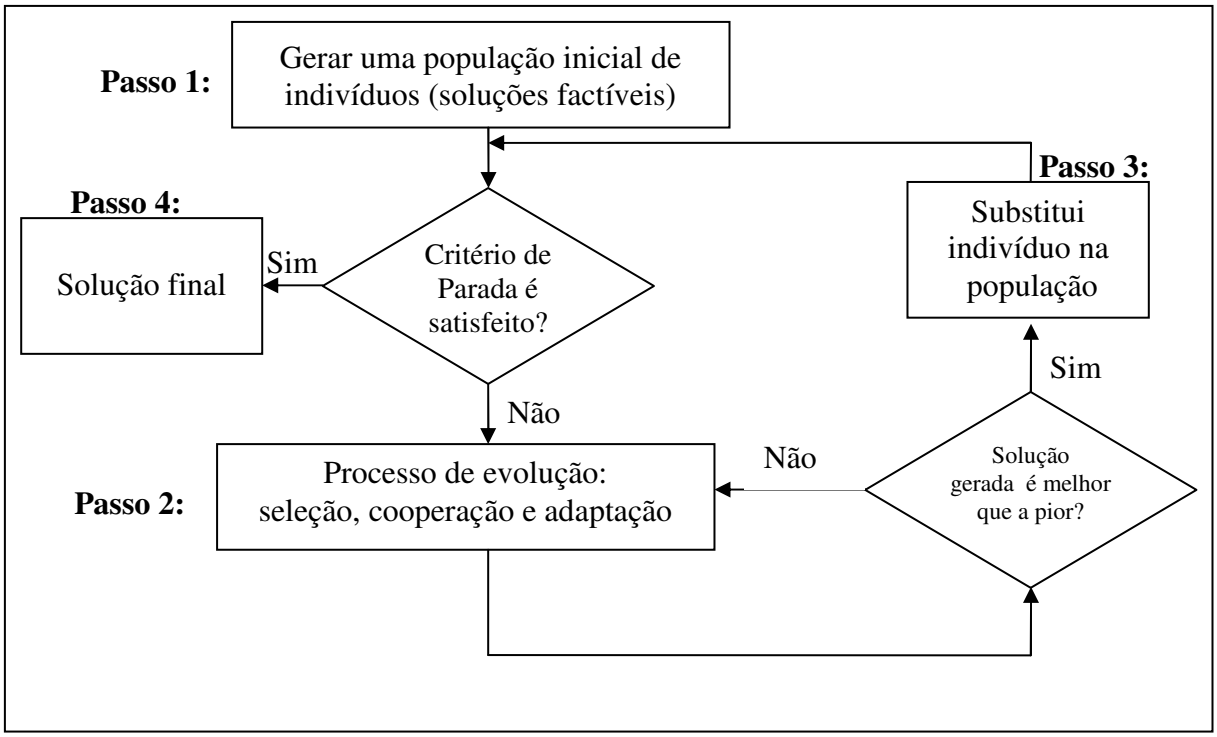


Figura 8 - Heurística proposta para o problema de corte unidimensional inteiro.

A seguir descreveremos com mais detalhes cada um dos procedimentos utilizados. Para uma visão geral sobre algoritmos evolutivos veja o Apêndice.

3.1.1 Representação da solução

Considere um indivíduo[3] como sendo uma solução factível para o problema de corte de estoque, tal que seus padrões de corte podem ser caracterizados como um cromossomo[4], cujos genes[5] representem cada um destes padrões de corte, seu respectivo objeto e a quantidade de vezes que o padrão será utilizado (x_{jk}) (Figura 9). Onde a_{jk} é o vetor correspondente ao *j-ésimo* padrão de corte sobre o objeto k, $j=1,...,Nk$, $k=1,...,K$.

[3] Indivíduo representa a solução para o sistema, pode ser formado por um ou mais cromossomos.

[4] Cromossomo é formado por um conjunto de genes.

[5] Gene representa ou controla uma característica específica de um indivíduo.

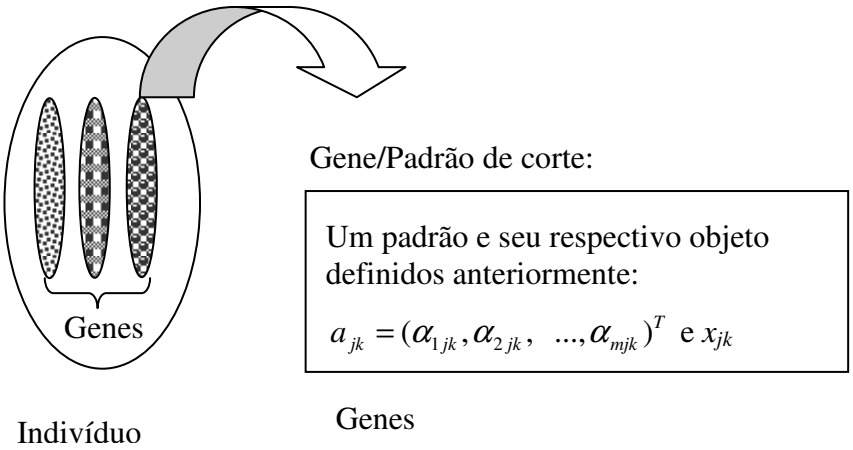


Figura 9 - Representação de um indivíduo.

Um exemplo de indivíduo é mostrado na Tabela 3.

Tabela 3 - Indivíduo gerado.

Número do objeto	x_{jk}	**Padrão (a_{jk})**		
1	4	0	2	1
1	3	1	0	2
2	6	1	2	0

3.1.2 População inicial

A população inicial é um conjunto de várias soluções factíveis (conjunto de indivíduos). O tamanho da população inicial é um parâmetro a ser estabelecido no algoritmo evolutivo. A geração da população inicial será feita utilizando-se de diferentes adaptações do método de repetição exaustiva, nos quais outras heurísticas, além da FFD, serão utilizadas.

3.1.2.1 Heurística de repetição exaustiva 1 (HRE1)

A seguir, para facilitar a leitura, repetimos a heurística de repetição exaustiva utilizando FFD apresentada na seção 2.4.1.2.1:

Algoritmo Heurístico de Repetição Exaustiva utilizando FFD (HRE1)

Início do algoritmo

Declaração das váriavies

*m:*quantidade de itens
K: quantidade de objetos em estoque
L_k*:* comprimento do objeto *k, k=1,...,K*
l_i*:* comprimento do item *i*, *i=1,...,m*
d_i*:* demando do item *i, i=1,...,m*
r_i: demanda residual do item *i* será a demanda do item *i* (d_i)
D: será a demanda residual total que consiste do somatório da demanda de cada item *i*
e_k*: estoque do objeto k, k=1,...,K*
s_k: estoque residual do objeto *k* será o estoque do objeto *k* (e_k), *k=1,...,K*
j: número do padrão de corte para o objeto *K*
RESTO: comprimento restando do objeto *k*
α_{ik} : quantidade de vezes que o item *i* será utilizado em relação ao objeto *k*
w_k*:* armazena a perda em relação ao objeto *k, k=1,...,K*
p: menor perda encontrada entre os objetos *k, k=1,...,K*
x_{jp}*:* quantidade de vezes que o padrão j com a menor perda será utilizado

Passo 1: ordene os itens a serem demandados, segundo o seu comprimento, em ordem decrescente.

$$l_1 \geq l_2 \geq ... \geq l_m$$

Faça $r_i = d_i \; i = 1,...,m; s_k = e_k \; k = 1,...,K; \; j = 1; D = \sum_{i=1}^{m} r_i$

Passo 2: *Enquanto* $D > 0$ *(enquanto existir demanda a ser satisfeita) faça*

Para $k = 1,...,K$

se $s_k > 0$

Faça $i = 1; RESTO = L_k$

Enquanto $i \leq m$ e $RESTO \geq l_m$ faça:

$$\alpha_{ik} = \min\left\{\left\lfloor \frac{RESTO}{l_i} \right\rfloor, r_i\right\}$$

RESTO = RESTO - $\alpha_{ik} * l_i$

i = i + 1

{fim enquanto}

{fim se}

$w_k = l_k - RESTO$

{fim para}

{fim enquanto}

Passo 3: Determine p tal que: $w_p = \min\{w_k, \text{tal que } s_k > 0, \text{para } k = 1,...,K\}$

Determine o número de vezes que cada padrão será utilizado:

$$x_{jp} = \min\left\{\left\lfloor \frac{r_i}{\alpha_{ip}} \right\rfloor, s_p, \text{onde } \alpha_{ip} \neq 0, \text{para } i = 1,...,m\right\}$$

Passo 4: {atualização}

$r_i = r_i - x_{jp} * \alpha_{ip}$, para $i = 1,...,m$

$D = D = -\sum_{i=1}^{m} x_{jp}(\alpha_{ip})$

$s_p = s_p - x_{jp}$, para $i = 1,...,m$

$j = j + 1$

{fim enquanto (Passo 2)}

Fim do algoritmo.

Observa-se que, ao utilizar este procedimento, apenas um indivíduo é gerado.

3.1.2.2 Heurística de repetição exaustiva 2 (HRE2)

Este método é semelhante à heurística de repetição exaustiva utilizando FFD (HR1) com alterações nos **Passos 2**. no qual a escolha do item será de forma aleatória. O algoritmo adaptado será visualizado a seguir.

Início do algoritmo

Passo 1: ordenação dos itens igual seção à 2.4.1.2.1.

Passo 2: Enquanto $D > 0$ (enquanto existir demanda a ser satisfeita) faça

Para $k = 1,...,K$

se $s_k > 0$

$RESTO = L_k$

Escolha aleatoriamente um item entre 1 e m tal que $r_i > 0 \, e \, l_i < L_K$

Se $RESTO \geq l_m$ então

$$\alpha_{ik} = \min\left\{\left\lfloor \frac{RESTO}{l_i} \right\rfloor, r_i\right\}$$

RESTO = RESTO - $\alpha_{ik} * l_i$

{fim se}

{fim se}

$w_k = l_k - RESTO$

{fim para}

{fim enquanto}

Passo 3: igual à seção 2.4.1.2.1.

Passo 4: igual à seção 2.4.1.2.1.

Fim do algoritmo.

Observa-se que, a aplicação deste procedimento várias vezes gera indivíduos diferentes.

3.1.2.3 Heurística de repetição exaustiva 3 (HRE3)

Esta heurística é similar à heurística de repetição exaustiva (HRE1), utilizando FFD com alteração no **Passo 2**, no qual a escolha dos objetos é aleatória. O padrão escolhido será o primeiro padrão construído para aquele objeto, não sendo verificada a perda do padrão, modificando assim o **Passo 3**. A seguir, é demonstrado o algoritmo modificado.

Início do algoritmo

Passo 1: ordenação dos itens igual à seção 2.4.1.2.1.

Passo 2: Enquanto $D > 0$ (enquanto existir demanda a ser satisfeita) faça

Escolhe aleatoriamente um objeto p entre 1 e K tal que $s_k > 0$

Faça $i = 1; RESTO = L_k$

Enquanto $i \leq m$ e $RESTO \geq l_m$ faça:

$$\alpha_{ik} = \min\left\{\left\lfloor \frac{RESTO}{l_i} \right\rfloor, r_i\right\}$$

$RESTO = RESTO - \alpha_{ik} * l_i$

$i = i + 1$

{fim enquanto}

{fim para}

{fim enquanto}

Passo 3: Determine o número de vezes que cada padrão será utilizado:

$$x_{jp} = \min\left\{\left\lfloor \frac{r_i}{\alpha_{ip}} \right\rfloor, s_p, \text{onde } \alpha_{ip} \neq 0, \text{para } i = 1,...,m\right\}$$

Passo 4: igual à seção 2.4.1.2.1.

Fim do algoritmo.

Aqui também é possível gerar vários indivíduos diferentes. Além disso tais indivíduos são diferentes dos indivíduos gerados anteriormente.

3.1.2.4 Heurística de repetição exaustiva 4 (HR4)

Esta heurística é similar ao método que utiliza usa uma heurística de construção de repetição exaustiva (HRE1), utilizando FFD com alterações nos **Passos 2** e **3** do algoritmo apresentado na seção 2.4.1.2.1 No **Passo 2** a escolha dos objetos e dos itens será de forma aleatória, já no **Passo 3**, não é determinada a perda de cada padrão, de maneira que, o primeiro padrão construído será utilizado. O algoritmo modificado é apresentado a seguir:

Início do algoritmo

Passo 1: igual seção 2.4.1.2.1.

Passo 2: Enquanto $D > 0$ (enquanto existir demanda a ser satisfeita) faça

Escolhe aleatoriamente um objeto p entre 1 e K tal que $s_k > 0$

$RESTO = L_k$

Escolhe aleatoriamente um item entre 1 e m tal que $r_i > 0 \, e \, l_i < l_K$

Se $RESTO \geq l_m$ então: $\alpha_{ik} = \min\left\{\left\lfloor \frac{RESTO}{l_i} \right\rfloor, r_i\right\}$

$RESTO = RESTO - \alpha_{ik} * l_i$

{fim se}

{fim enquanto}

Passo 3: Determine o número de vezes que cada padrão será utilizado:

$$x_{jp} = \min\left\{\left\lfloor \frac{r_i}{\alpha_{ip}} \right\rfloor, s_p, \text{onde } \alpha_{ip} \neq 0, \text{para } i = 1,...,m\right\}$$

Passo 4: igual à seção 2.4.1.2.1.

Fim do algoritmo.

Os indivíduos obtidos com este procedimento são diferentes dos indivíduos gerados anteriormente, podendo-se gerar vários indivíduos com esta heurística.

3.1.2.5 Considerações sobre os métodos de geração

Os cinco métodos de geração da população inicial apresentados nas seções anteriores foram desenvolvidos para gerar uma população inicial num rápido tempo computacional e com diversidade de indivíduos. A diversidade de indivíduos é fundamental para garantir a busca de melhores soluções no algoritmo evolutivo proposto neste trabalho.

A Tabela 4, a seguir, demonstra a forma como as heurísticas de repetição exaustiva foram desenvolvidas baseadas nos seguintes critérios: forma de ordenação dos itens, escolha do objeto de menor perda e a maneira de escolha dos objetos e itens.

Tabela 4 - Descrição dos métodos de repetição exaustivos desenvolvidos.

Característica / Método	Ordenação dos itens	Escolha do objeto	Escolha do item	Verifica perda?
HRE1	Decrescente de comprimento	De *1* até *K*	Conforme ordenação dos itens, do maior para o menor.	sim
HRE2	Decrescente de comprimento	De *1* até *K*	Aleatória.	sim
HRE3	Decrescente de comprimento	Aleatória	Conforme ordenação dos itens, do maior para o menor.	não
HRE4	Decrescente de comprimento	Aleatória	Aleatória	não

A Tabela 5 mostra a quantidade de indivíduos gerados, utilizando cada um dos métodos apresentados anteriormente. A heurística de repetição exaustiva HRE1 gera apenas um indivíduo enquanto as demais (HRE2, HRE3 e HRE4) os indivíduos são obtidos por um percentual em relação ao tamanho da população inicial. O percentual de cada um dos métodos é um dos parâmetros a serem estabelecidos no algoritmo evolutivo proposto.

Tabela 5 - Quantidade de indivíduos gerados para cada método.

Método	Quantidade de Indivíduos
HRE1	1
HRE2	%
HRE3	%
HRE4	%

3.1.3 Algoritmo Evolutivo e Nova solução gerada

Nesta seção serão descritos os detalhes do algoritmo evolutivo utilizado no desenvolvimento deste trabalho, tais como: processo de seleção, cooperação e adaptação.

Algumas inicializações são realizadas antes de aplicar o algoritmo evolutivo:

- ordenar a população inicial em ordem crescente do valor de avaliação (*fitness*)
- *g*: número de genes de um indivíduo (será abortado na seção 3.1.3.2)
- *n_padrao* número inicial do padrão de corte de um indivíduo, inicialmente 0 (zero)
- r_i: demanda residual do item *i* será a demanda do item *i* (d_i)

- D: será a demanda residual total que consiste do somatório da demanda de cada item i
- s_k: a estoque residual do objeto k será o estoque do objeto k (e_k)

3.1.3.1 Processo de seleção

O processo de seleção será realizado para escolher um indivíduo que gerará uma nova solução. A seleção de um indivíduo será de forma gulosa/aleatória, isto é, indivíduos melhores (função de avaliação) terão probabilidade maior de serem selecionados. A população de indivíduos é ordenada de forma crescente do valor da função de avaliação, facilitando, assim, o processo de seleção do indivíduo. A probabilidade de seleção de um indivíduo será um dos parâmetros a serem determinados no algoritmo evolutivo.

3.1.3.2 Cooperação

Após a escolha do indivíduo, utilizando o processo de seleção, um padrão de corte, para o indivíduo que está sendo gerado, deverá ser selecionado. A forma de seleção do padrão de corte de um indivíduo será aleatória.

O padrão de corte selecionado poderá ou não ser inserido no novo indivíduo, pois o mesmo pode ser descartado quando se referir a um objeto que não possui mais quantidade em estoque, ou quando existir um item, neste padrão de corte, no qual a demanda residual já foi atendida. Nos demais casos serão calculados a quantidade de vezes que o padrão será utilizado e inseridos no novo indivíduo.

O processo continuará por meio da seleção de um outro indivíduo e um novo padrão de corte.

O procedimento será finalizado quando a demanda residual total for atendida ou quando o número máximo de tentativas de inserções de padrões de cortes no novo indivíduo for alcançado (parâmetro g). Este parâmetro é definido baseado nas características da população inicial, sendo a média do número de padrões de cortes gerados de todos os indivíduos na população inicial mais um percentual pré-estabelecido.

Se a demanda residual total dos itens for zero um novo indivíduo factível foi gerado para a nova população. Caso contrário é necessário uma fase de adaptação.

3.1.3.3 Adaptação

Sendo a demanda residual dos itens diferente de zero, após *g* tentativas, a construção de uma solução factível será necessária. A construção será utilizando a heurística de repetição exaustiva utilizando FFD (apresentada na seção 2.4.1.2.1), considerando apenas a demanda residual dos itens.

3.1.3.4 Substituição do indivíduo na população

O novo indivíduo encontrado será substituído na população utilizando a estratégia de substituição populacional *Steady-State* (definido no Apêndice), que consiste na substituição do melhor indivíduo pelo pior indivíduo de todos. Lembrando que a população inicial está ordenada de forma crescente em relação à função de avaliação de cada indivíduo. Após a ordenação dos indivíduos (em ordem crescente da função de avaliação) a inserção da nova solução gerada encontrará a posição correta, na população de indivíduos, verificando o novo valor da função de avaliação, obtido para o indivíduo gerado, se é menor ou igual a qualquer outro valor de avaliação dos indivíduos que compõem a população. Se o valor da função de avaliação for igual um segundo critério de desempate será utilizado, neste caso o menor número de padrões de corte do indivíduo.

Os demais indivíduos são deslocados para a direita, eliminando o pior de todos (última posição da população).

3.1.3.5 Critério de parada e solução final

O critério de parada é mais um parâmetro a ser definido no algoritmo evolutivo. Este critério é o número máximo de iterações, no qual cada iteração representa a geração de um indivíduo. Se o critério de parada for atendido a solução final é determinada selecionando-se o primeiro elemento da população de indivíduos. O indivíduo que está na primeira posição será o indivíduo que possui a melhor função de avaliação entre os demais, pois a população está ordenada de forma crescente em relação ao valor de avaliação (*fitness*) de cada indivíduo. Caso não seja satisfeito o critério de parada, passará para o processo de evolução (Figura 8).

3.1.3.6 Função de avaliação (*fitness*)

A função de avaliação (*fitness*) utilizada neste trabalho leva em consideração dois aspectos (descritos na seção 2.2): o total de perda do objeto e o número de padrões de corte. A função de avaliação 3.1 mostrada a seguir poderá ser avaliada dando mais importância a algum aspecto. Para isso determinados os parâmetros: $alfa_1$ e $alfa_2$ que são os percentuais de avaliação que será verificado para cada um dos aspectos. Estes parâmetros devem ser determinados no algoritmo evolutivo.

$$\text{Minimizar } f(x) = \mathbf{alfa_1}\left(\frac{\sum_{k=1}^{K}\sum_{j=1}^{N_K}(L_k - \sum_{i=1}^{m} l_i \alpha_{ijk})x_{jk}}{\sum_{k=1}^{K} L_k e_k}\right) + \mathbf{alfa_2}\left(\frac{\sum_{k=1}^{K}\sum_{j=1}^{N_k}\delta(x_{jk})x_{jk}}{\sum_{k=1}^{K} e_k}\right) \quad (3.1)$$

Observe que os denominadores de cada termo são utilizados para normalizar seus valores entre $[0,1]$ e o melhor indivíduo será aquele que obtiver o menor valor para a sua função de avaliação.

3.1.3.7 Parâmetros evolutivos a serem estabelecidos

O desempenho do algoritmo evolutivo é fortemente influenciado pela definição dos parâmetros a serem utilizados. A seguir são apresentados os parâmetros utilizados, bem como a simbologia adotada no algoritmo evolutivo detalhado, a ser apresentado na seção 3.1.3.8.

- *tamanho da população inicial: POPULACAO_INICIAL ;*
- *percentual das adaptações: h2, h3 e h4 para a geração da população inicial;*
- *percentual da seleção gulosa/aleatória de um indivíduo: PERCENTUAL;*
- *número de padrões de corte (gene) de um indivíduo: g;*
- *critério de parada: número máximo de interações (gerações da população): Max_iteracoes;*
- *função de avaliação (fitness): f.*

3.1.3.8 Algoritmo evolutivo detalhado

A Figura 13 mostra o algoritmo evolutivo detalhado proposto. Uma população inicial de indivíduos é gerada com soluções factíveis (conjunto de indivíduos factíveis). Em geral, na primeira iteração o critério de parada não será satisfeito. Começa-se o processo de evolução (seleção, cooperação e adaptação). Na seleção será escolhido o indivíduo conforme o que foi descrito na seção 3.1.3.1. No processo de cooperação as informações dos indivíduos são trocadas entre si. O novo indivíduo gerado poderá ou não ser factível. Se o novo indivíduo não for factível o processo de adaptação será executado até encontrar um indivíduo factível. Verifica-se se a nova solução encontrada é melhor do que a pior solução da população. O novo indivíduo gerado será substituído na população e o pior indivíduo desta será eliminado. Se o critério de parada não for satisfeito o processo de reprodução, cooperação e adaptação, será repetido, senão uma solução final, factível, é encontrada.

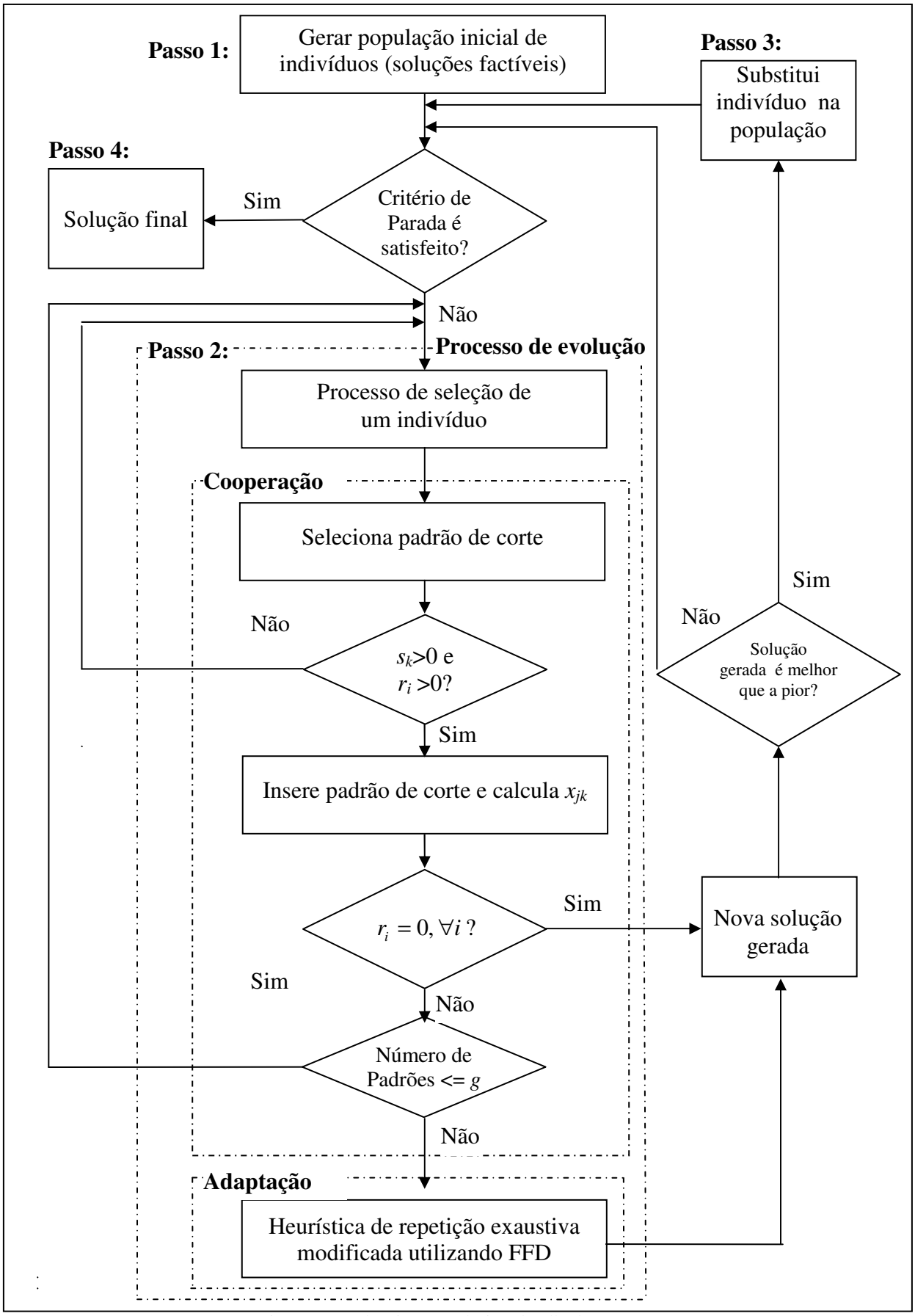


Figura 10 - Algoritmo evolutivo detalhado.

Apresentamos o algoritmo evolutivo com um dos conjuntos de parâmetros utilizados nos testes computacionais.

Algoritmo evolutivo (AE)

Parâmetros do AE

POPULACAO_INICIAL = 10; *//número de indivíduos da população inicial*

Max_Iteracoes = 4500; *//número máximo de iterações (gerações)*

PERCENTUAL = 50; *//percentual de escolha gulosa/aleatória dos melhores //indivíduos*

//A função de avaliação é definida conforme a ênfase que se quer dar a alguns aspectos: //perda, número de objetos cortados e número de padrões de corte

$alfa_1 = 0.33$; *// percentual de importância da perda a ser considerada na função de avaliação*

$alfa_2 = 0.33$; *// percentual de importância do número de objetos a ser considerada na função //de avaliação*

$alfa_3 = 0.33$; *// percentual de importância do número de padrões de cortes considerado na //função avaliação*

//O percentual de geração da população inicial através das heurísticas de repetição exaustiva //adaptadas será:

h1 = 0.33; *// 33 % dos indivíduos serão gerados utilizando a adaptação h1*

h2 = 0.33; *// 33 % dos indivíduos serão gerados utilizando a adaptação h2*

h3 = 0.33; *// 33 % dos indivíduos serão gerados utilizando a adaptação h3*

Fim da declaração dos parâmetros do A.E.

Variáveis

D; // demanda residual total a ser atendida

n; //número atual de genes (padrões de corte)

count; //contador atual do número de iterações atuais

f; //valor da função de avaliação

g; //número máximo de genes inseridos

S; //indivíduo gerado a cada iteração

$\bar{S}$ *; //melhor indivíduo da solução*

população; //população de indivíduos

Fim da declaração das variáveis

Início do algoritmo

Passo 1 *// Inicialize as variáveis e construa a população inicial*

Faça $S = \phi$, $f = 0$ e $populacao = \{S_1, S_2, S_3, S_4, S_5, S_6, S_7, S_8, S_9, S_{10}\}$

//S_1 a S_{10} são indivíduos que contém as soluções obtidas por: HRE-FFD e adaptações //(HRE1,HRE2,HRE3 e HRE4) descritos na seção 3.1.2

Faça *count = 0 / /iteração atual*

/ /Inicializa a demanda e o estoque inicial

Faça $r_i = d_i$ para $i = 1,..,.m$ e $s_k = e_k$ para $k=1,...,K$ *//r_i e s_k são respectivamente a demanda //residual do item i e o estoque residual do objeto k*

$D = \sum_{i=1}^{m} d_i$ *//demanda residual total será o somatório da demanda residual de cada item*

Fim Passo 1

Passo 2 *// Processo de evolução – construção da população*

Faça: seleção gulosa/aleatória de um indivíduo entre S_1 e S_{10} em *população //na seleção é //dada à preferência a indivíduos com valor de avaliação melhor (veja seção3.1.3.6)*

Passo 2.1 *//* Cooperação*: seleção aleatória de um padrão e inserção dele no novo indivíduo*

Faça: escolha aleatoriamente de um padrão $a_{jk} = [\alpha_{1jk}, \alpha_{2jk}, ..., \alpha_{mjk}]^T$ do indivíduo selecionado

Se $s_k > 0$ e $r_i > 0$ para $i = 1,...,m$ então *//se existir objeto em estoque e demanda do item*

// Determine o número de vezes que o padrão será utilizado

Faça: $x_{jk} = \min\left\{\left\lfloor \frac{r_i}{\alpha_{ijk}} \right\rfloor, er_j\right\} \alpha_{jki} \neq 0$, para $i = 1...m$ e $j =$ barra do padrão

Faça: $S = S \cup \{a_{jk}, x_{jk}\}$ *// insere padrão no novo indivíduo (reprodução)*

// atualiza a demanda e o estoque

Faça: $r_i = r_i - x_{jk}\alpha_{ijk}$ para $i = 1...m$ e $s_k = s_k - x_{jk}$ e $j =$ barra do padrão

Faça: $n = n + 1$

Senão volte ao Passo 2 *//seleciona outro indivíduo*

Se $D = 0$ então // se a demanda residual total for igual a zero uma nova solução foi gerada

Senão

se $n < g$ ou $D > 0$, volte ao passo 2 *// número de padrões (n) não for igual ao //número máximo de padrões (g) e ou a demanda residual total (D) não seja //igual a zero*

Senão passo 2.2

Fim Passo 2.1

Passo 2.2 *// Adaptação:teste de otimalidade da solução, término da solução e atualização da //população*

Se $D \neq 0$ então aplicar a heurística de repetição exaustiva utilizando FFD *//considerando a //demanda residual gera uma solução*

$f = F(S)$ //F calcula o valor de avaliação de um indivíduo

Fim Passo 2.2

Fim do Passo 2

Início do Passo 3

Se $f < F(S_k)$ para $k = 1...POPULACAO\_INICIAL$ então $S_k = S$ *//A substituição de um indivíduo será feita conforme o critério da seção 3.1.3.4*

Faça: *count = count + 1*

Se *count = Max_iteracoes* então Passo 4

Senão volte ao Passo 2

Fim do Passo 3

Passo 4*//Solução final: substituição do indivíduo na população*

Faça: $\bar{S} = poulacao\{S_{melhor}\}$ *//o melhor indivíduo da população é escolhido*

Fim do algoritmo

Fim Passo 4

Fim do algoritmo

3.1.4 Exemplo utilizando Algoritmo Evolutivo

Nesta seção apresentamos um exemplo de solução para o problema de corte utilizando o Algoritmo Evolutivo apresentado na seção 3.1.3.8.

3.1.4.1 Geração de uma população inicial (Passo 1 do A.E)

A Tabela 6 mostra os dados de entrada dos objetos (a) e itens (b), exemplo este que faz parte do conjunto de exemplos práticos executados neste trabalho.

Tabela 6 - Dados de entrada.

Objetos

Número do objeto	Comprimento	Estoque
1	137	12
2	706	2
3	589	15

(a)

Itens

Número do item	Comprimento	Demanda
1	29	**5**
2	61	**7**
3	71	**1**
4	15	**2**
5	25	**3**

(b)

O tamanho da população inicial, neste exemplo, será de 10 indivíduos, e o percentual de indivíduos gerados por meio das heurísticas de repetição exaustiva será de 33% do tamanho da população inicial, descontado o indivíduo gerados pela heurística de repetição exaustiva: HRE1. A quantidade de indivíduos gerados em cada uma das heurísticas de repetição exaustivas é visualizada na Tabela 7.

Tabela 7 - Quantidade de indivíduos gerados na população inicial.

Método	Quantidade de Indivíduos
HRE1	1
HRE2	**3**
HRE3	**3**
HRE4	**3**
Total	**10**

O primeiro indivíduo é gerado pela a heurística de repetição exaustiva, utilizando FFD (HRE1), com perda total de 115, 3 padrões de corte, 3 objetos cortados e a função de avaliação (apresentada na seção 3.1.3.6, considerando $alfa_1=0,5$ e $alfa_2=0,5$) igual a 0,056560 (Tabela 8).

Tabela 8 - Indivíduo gerado com a HRE1.

Número do objeto	**x**	**Perda**	**Padrão**				
3	1	4	3	7	1	0	0
1	1	4	2	0	0	0	3
1	1	107	0	0	0	2	0
Demanda			**5**	**7**	**1**	**2**	**3**

A Tabela 9 mostra o segundo indivíduo gerado, utilizando a heurística de repetição exaustiva 1 (HRE1), sendo o primeiro desta heurística, com perda total de 704, 4 padrões de corte, 4 objetos cortados e a função de avaliação igual a 0,098568.

Tabela 9 – Primeiro indivíduo gerado com a HRE2.

Número do objeto	**x**	**Perda**	**Padrão**				
3	1	7	5	6	1	0	0
1	1	62	0	0	0	0	3
3	1	528	0	1	0	0	0
1	1	107	0	0	0	2	0
Demanda			**5**	**7**	**1**	**2**	**3**

A Tabela 10 mostra o terceiro indivíduo gerado utilizando a heurística de repetição exaustiva 2 (HRE2), sendo o segundo desta heurística, com perda total de 74, 4 padrões de corte, 6 objetos cortados e a função de avaliação igual a 0,072077.

Tabela 10 - Segundo indivíduo gerado com HRE2.

Número do objeto	**x**	**Perda**	**Padrão**				
1	3	45	0	2	0	0	0
1	1	6	4	0	0	1	0
1	1	1	0	1	0	0	3
1	1	22	1	0	1	1	0
Demanda			**5**	**7**	**1**	**2**	**3**

A Tabela 11 mostra o quarto indivíduo gerado utilizando a heurística de repetição exaustiva 3 (HRE3), sendo o terceiro desta heurística, com perda total de 704, 4 padrões de corte, 4 objetos cortados e a função de avaliação igual a 0,098568.

Tabela 11 - Terceiro indivíduo gerado com a HRE2.

Número do objeto	**x**	**Perda**	**Padrão**				
1	1	4	2	0	0	0	3
1	1	20	3	0	0	2	0
3	1	162	0	7	0	0	0
3	1	518	0	0	1	0	0
Demanda			**5**	**7**	**1**	**2**	**3**

A Tabela 12 mostra o quinto indivíduo gerado utilizando a heurística de repetição exaustiva 3 (HRE3), sendo o primeiro desta heurística, com perda total de 567, 3 padrões de corte, 3 objetos cortados e a função de avaliação igual a 0,075566.

Tabela 12 - Primeiro indivíduo gerado com a HRE3.

Número do objeto	**X**	**Perda**	**Padrão**				
3	1	4	3	7	1	0	0
1	1	4	2	0	0	0	3
3	1	559	0	0	0	2	0
Demanda			**5**	**7**	**1**	**2**	**3**

A Tabela 13 mostra o sexto indivíduo gerado utilizando a heurística de repetição exaustiva 3 (HRE3), sendo o segundo desta heurística, com perda total de 547, 2 padrões de corte, 2 objetos cortados e a função de avaliação igual a 0,0,057483.

Tabela 13 – Segundo indivíduo gerado com a HRE3.

Número do objeto	**x**	**Perda**	**Padrão**				
3	1	4	3	7	1	0	0
2	1	543	2	0	0	2	3
Demanda			**5**	**7**	**1**	**2**	**3**

A Tabela 14 mostra o sétimo indivíduo gerado utilizando a heurística de repetição exaustiva 3 (HRE3), sendo o terceiro desta heurística, com perda total de 664, 2 padrões de corte, 2 objetos cortados e a função de avaliação igual a 0,062403.

Tabela 14 - Terceiro indivíduo gerado com HRE3.

Número do objeto	**x**	**Perda**	**Padrão**				
2	1	13	5	7	1	0	2
2	1	651	0	0	0	2	1
Demanda			**5**	**7**	**1**	**2**	**3**

A Tabela 15 mostra o oitavo indivíduo gerado utilizando a heurística de repetição exaustiva 4 (HRE4), sendo o primeiro desta heurística, com perda total de 1273, 4 padrões de corte, 4 objetos cortados e a função de avaliação igual a 0,0,122493.

Tabela 15 – Primeiro indivíduo gerado com HRE4.

Número do objeto	**X**	**Perda**	**Padrão**				
2	1	530	0	0	1	2	3
1	1	21	4	0	0	0	0
3	1	560	1	0	0	0	0
3	1	162	0	7	0	0	0
Demanda			**5**	**7**	**1**	**2**	**3**

A Tabela 16 mostra o nono indivíduo gerado utilizando a heurística de repetição exaustiva 4 (HRE4), sendo o segundo desta heurística, com perda total de 1979, 5 padrões de corte, 5 objetos cortados e a função de avaliação igual a 0,169421.

Tabela 16 - Segundo indivíduo gerado com HRE4.

Número do objeto	**x**	**Perda**	**Padrão**				
2	1	601	0	0	0	2	3
2	1	279	0	7	0	0	0
1	1	21	4	0	0	0	0
3	1	560	1	0	0	0	0
3	1	518	0	0	1	0	0
Demanda			**5**	**7**	**1**	**2**	**3**

A Tabela 17 mostra o décimo e último indivíduo gerado utilizando a heurística de repetição exaustiva 4 (HRE4), sendo o segundo desta heurística, com perda total de 1664, 6 padrões de corte, 6 objetos cortados e a função de avaliação igual a 0,173417.

Tabela 17 - Terceiro indivíduo gerado com HRE4.

Número do objeto	**x**	**Perda**	**Padrão**				
1	1	5	0	1	1	0	0
2	1	676	0	0	0	2	0
2	1	631	0	0	0	0	3
3	1	223	0	6	0	0	0
1	1	21	4	0	0	0	0
1	1	108	1	0	0	0	0
Demanda			**5**	**7**	**1**	**2**	**3**

O número de padrões e o valor da função de avaliação (*fitness*), da população inicial de tamanho dez, é visualizada na Tabela 18.

Tabela 18 - População inicialmente formada.

Número do indivíduo	**Número de padrões de corte**	**Perda total**	**Objetos cortados**	**Função de avaliação (*fitness*)**
1	3	115	3	0,056560
2	4	704	4	0,098568
3	4	74	6	0,072077
4	4	704	4	0,098568
5	3	567	3	0,075566
6	2	547	2	0,057483
7	2	664	2	0,062403
8	4	1273	4	0,122493
9	5	1979	5	0,169421
10	6	1664	6	0,173417

3.1.4.2 Processo de evolução (Passo 2 do A.E.)

Inicialmente, a população inicial obtida, na seção anterior, deve ser ordenada de forma crescente do valor de avaliação (indivíduos melhores são aqueles que apresentam o menor valor na sua função de avaliação). A Tabela 19 apresenta a população inicial ordena.

Tabela 19 - População inicial ordenada.

Número do indivíduo	Posição do indivíduo antes da ordenação	Número de padrões de corte	Perda total	Objetos cortados	Função de avaliação (*fitness*)
1	1	3	115	3	0,056560
2	6	2	547	2	0,057483
3	7	2	664	2	0,062403
4	3	4	74	6	0,072077
5	5	3	567	3	0,075566
6	2	4	704	4	0,098568
7	4	4	704	4	0,098568
8	8	4	1273	4	0,122493
9	9	5	1979	5	0,169421
10	10	6	1664	6	0,173417

Uma segunda atribuição faz-se necessário: o número de genes de um novo (parâmetro *g*). Neste exemplo, a média geral dos padrões da população inicial (Tabela 19) é de 3,7 por indivíduo. Acrescentando 20% sobre a média (parâmetro a ser estabelecido) resultará no valor de $g = 5$ (arredondado para o inteiro superior mais próximo).

Após a atribuição de *g*, inicia-se o processo de seleção de um indivíduo da população ordenada (conforme seção 3.1.3.1). O primeiro indivíduo selecionado, neste exemplo, é o décimo da Tabela 19, o mesmo é visualizado na Tabela 20 a seguir.

Tabela 20 – Primeiro indivíduo selecionado.

Número do objeto	x	Perda	Padrão				
1	1	5	0	1	1	0	0
2	1	676	0	0	0	2	0
2	1	631	0	0	0	0	3
3	1	223	0	6	0	0	0
1	1	21	4	0	0	0	0
1	1	108	1	0	0	0	0
Demanda			**5**	**7**	**1**	**2**	**3**

Após o processo de seleção de um indivíduo inicia-se o processo de cooperação (seção 3.1.3.2). Neste processo o primeiro padrão de corte é selecionado (padrão de corte destacado na Tabela 20 na cor cinza) e inserido no novo indivíduo. Calcula-se a quantidade máxima de vezes que o mesmo será utilizado, neste caso apenas uma vez.

O procedimento de seleção de um outro indivíduo e o seu padrão de corte é repetido até que o número de padrões de corte selecionado seja maior que g. O indivíduo obtido até este momento é visualizado na Tabela 21.

Tabela 21 - Indivíduo gerado no início do passo 2 do A.E.

Número do objeto	**x**	**Perda**	**Padrão**				
1	1	5	0	1	1	0	0
1	1	20	3	0	0	2	0
Demanda total por item a ser atendida			**5**	**7**	**1**	**2**	**3**
Demanda residual (demanda a ser atendida)			**2**	**6**	**0**	**0**	**3**

A Tabela 22 mostra o primeiro indivíduo gerado, após o processo de adaptação (seção 3.1.3.3), com perda total de 74, 4 padrões de corte, 6 objetos cortados e a função de avaliação igual a 0,072077.

Tabela 22 – Nova solução gerada com o A.E.

Número do objeto	**x**	**Perda**	**Padrão**				
1	1	5	0	1	1	0	0
1	1	20	3	0	0	2	0
1	3	45	0	2	0	2	0
1	1	4	2	0	0	0	3
Demanda			**5**	**7**	**1**	**2**	**3**

3.1.4.3 Substituição do indivíduo na população (passo 3 do A.E.)

A nova solução encontrada, indivíduo da Tabela 22, após o processo de evolução, será substituído na população inicial (Tabela 19) conforme o processo de substituição de um indivíduo descrito na seção 3.1.4. A tabela 23 mostra a inserção da nova solução na população inicial, destacado na cor cinza.

Tabela 23 - População inicial após a inserção do novo indivíduo.

Número do indivíduo	Número de padrões de corte	Perda total	Objetos cortados	Função de avaliação (*fitness*)
1	3	115	3	0,056560
2	2	547	2	0,057483
3	2	664	2	0,062403
4	4	74	6	0,072077
5	**4**	**74**	**6**	**0,072077**
6	3	567	3	0,075566
7	4	704	4	0,098568
8	4	704	4	0,098568
9	4	1273	4	0,122493
10	5	1979	5	0,169421

3.1.4.4 Solução final (passo 4 do A.E.)

A melhor solução encontrada (indivíduo gerado), para este exemplo, após o critério de parada ser satisfeito (seção 3.1.3.5), com perda total de 115, 3 padrões de corte, 3 objetos cortados e a função de avaliação igual a 0,056560 é visualizado na Tabela 24.

Tabela 24 – Melhor solução encontrada ao termino do A.E.

Número do objeto	x	Perda	Padrão				
1	1	107	0	0	0	2	0
3	1	4	3	7	1	0	0
1	1	4	2	0	0	0	3
Demanda			**5**	**7**	**1**	**2**	**3**

4 IMPLEMENTAÇÃO E RESULTADOS COMPUTACIONAIS

Neste capítulo são apresentados o gerador aleatório utilizado para geração dos dados avaliados, os resultados computacionais obtidos e os resultados computacionais adicionais utilizando um segundo conjunto de exemplos.

O algoritmo evolutivo apresentado foi desenvolvido utilizando a linguagem C e a ferramenta C++ Builder 6. Os testes computacionais foram realizados utilizando um PC com processador Intel Pentium M (Centrino de 1.73 GHz) com 512 Mb de memória RAM.

4.1 Geração dos dados

Os dados utilizados para os testes foram obtidos por meio do gerador aleatório proposto por Wäscher e Gau (1995) e adaptados no trabalho de Poldi e Aranelas (2005) para o problema de corte em que se tem vários objetos em estoque (tamanhos diferentes).

As variações de tamanho do objeto, item, quantidade de estoque e demanda, descritas em Poldi (2003), são:

- número de objetos em estoque (*K*): 3, 5 e 7;
- número de tipos de itens (*m*): *5*, , 10, 20 e 40;
- dimensão dos objetos $L_k, k = 1,...,K$ são valores aleatórios no intervalo [1, 1000];
- estoques disponíveis dos objetos (e_k) foram gerados aleatoriamente no intervalo $\left[1,100\frac{m}{2}\right]$ onde *m* é o número de itens;
- tamanho dos itens $l_i, i = 1,...,m$ foram gerados aleatoriamente no intervalo [v_1L, v_2L], onde *L* é média simples da dimensão dos objetos. Utilizou-se *v1=0,01* e *v2=0,02* para gerar itens denominados *Pequenos* e *v2=0,08* para itens denominados *Médios*;

- demanda dos itens $d_i, i = 1,...,m$ são valores no intervalo [1,10].

O conjunto de exemplos, gerados e cedidos por Poldi (2003) está dividido em classes. No total são 18 classes, cada uma com 20 exemplos totalizando 360. As características das classes são visualizadas na Tabela 25.

Tabela 25 – Características das 18 classes.

Classe	Número de barras	Número de itens	Tamanho dos itens
C1	3	5	Pequeno
C2	3	5	Médio
C3	3	20	Pequeno
C4	3	20	Médio
C5	3	40	Pequeno
C6	5	40	Pequeno
C7	5	10	Pequeno
C8	5	10	Médio
C9	5	20	Pequeno
C10	5	20	Médio
C11	5	40	Pequeno
C12	5	40	Média
C13	7	10	Pequeno
C14	7	10	Médio
C15	7	20	Pequeno
C16	7	20	Médio
C17	5	40	Pequeno
C18	5	40	Médio

4.2 Resultados computacionais

Os resultados obtidos nesta seção são comparados com as heurísticas descritas nas seções anteriores: construtiva FFD (seção 2.4.1.2.1), construtiva Gulosa (seção 2.4.1.2.2), residual FFD (seção 2.4.1.3.2), residual Gulosa (seção 2.4.1.3.3), residual Nova 1 (seção 2.4.1.4.1), residual Nova 2 (seção 2.4.1.4.2), residual Nova 3 (seção 2.4.1.4.2) e algoritmo evolutivo (A.E.) (capítulo 3). A heurística proposta (A.E.) leva em consideração na

função objetivo a perda total e o número de padrões de corte, enquanto nas demais heurísticas avaliadas a perda é o objetivo principal.

Para os testes computacionais iniciais foram comparados dois parâmetros principais:

- perda total: soma das sobras no corte numa determinada solução;
- padrões de corte: número de padrões de corte diferentes necessários para o corte em uma solução;

Vários parâmetros evolutivos foram avaliados nos testes computacionais. Na Tabela 26 são apresentados um conjunto de parâmetros que se mostrou ser eficiente nos resultados obtidos (em termos de qualidade da solução e tempo computacional).

Tabela 26 - Parâmetros evolutivos utilizados.

Parâmetro	**Valor atribuído**
Tamanho da população inicial	10
Percentual de HRE2, HRE3 e HR4	33% cada
Percentual da seleção aleatória de um indivíduo	50% de probabilidade de seleção de um indivíduo em relação aos 50% melhores indivíduos da população
Número de genes (g)	A media do número de genes dos indivíduos da população inicial acrescido de 20%
Número máximo de interações (gerações)	1500
Função de avaliação ($alfa_1$ e $alfa_2$)	50% cada

A seguir são apresentados as comparações da perda e o número de padrões cortados com sete métodos heurísticos apresentados na seção (2.4.1).

A comparação da perda total é mostrada na Tabela 27. Os resultados obtidos foram bons. Segundo este critério o algoritmo evolutivo (A.E.), utilizando a Função B, obteve resultados melhores em 8 classes das 18 verificadas (44,94% dos resultados). No contexto geral o A.E. mostra-se melhor em relação às heurísticas avaliadas individualmente. Os melhores resultados estão destacados em negrito. Na Tabela 27 a coluna GG (Gilmore e Gomory) é a relaxação linear resolvida pelo método Simples com geração de colunas apresentado na seção 2.4.1.1.

Tabela 27 - Perda total para as 18 classes.

	Perda total								
		Repetição Exaustiva		**Residual**					
	GG	**FFD**	**Gulosa**	**FFD**	**Gulosa**	**Nova 1**	**Nova 2**	**Nova 3**	**A.E.**
C1	0,41	197,7	314,1	212,35	312,3	159,2	159,2	144,75	**118,3**
C2	251,65	1571,9	1161,1	686,25	796,9	445,4	**424,6**	426,25	520
C3	0	166,95	424,55	142,55	432,25	132,75	130,25	167,5	**39,85**
C4	33,85	814,15	6030,25	530,45	4438,25	**164,35**	208,7	171,8	349,5
C5	0	240,45	439,15	230,15	449,25	172,55	172,55	169,25	**70,35**
C6	10,7	447,05	4799	253,35	4885,65	**203,85**	230,5	229,1	324,5
C7	0	122,3	232,75	122,3	232,75	109,4	103,75	109,4	**58,2**
C8	190,81	979,3	2334,4	883,9	1306,4	308,9	386,3	**307,9**	411
C9	0	104,4	684,9	103,4	1579,55	101,45	106,3	105,05	**30,8**
C10	20,18	1906,35	5508,75	1892,9	3694,1	194,65	**155,45**	187,5	248,45
C11	0	134,15	127	129,05	80,1	136,9	132,2	137,05	**45,1**
C12	5,98	497,4	15716,75	16430,65	19507,8	**108,05**	144,1	107,85	333,35
C13	0	51,35	397,1	51,35	397,1	54,25	54,05	**50,75**	83,25
C14	23,4	1236,8	2368,25	678,75	1859	120,9	**116,95**	148,95	174,75
C15	0	110,05	286,8	99,15	290,15	92,1	88	92,05	**15**
C16	1,86	175,9	2838,5	423,15	1536,05	**81,7**	87,95	93,45	104,4
C17	0	91,25	634,7	81,85	632,45	68,2	66,8	73,6	**16,7**
C18	5,05	1594,45	12167,5	1327,45	6728,7	**112,8**	117,75	81,85	369,65
Total	543,89	10441,9	56465,55	24279	49158,75	2767,4	2885,4	**2804,05**	3313,15

A Tabela 28 mostra a comparação realizada com o critério do número de padrões de corte. O algoritmo evolutivo (A.E.) proposto obteve resultados melhores ainda em relação ao critério da perda. Das 18 classes avaliadas obtivemos resultados melhores em 17 (94% dos resultados). Apenas o resultado da classe C11 foi pior em relação a uma única heurística, a heurística Nova 3. Os melhores resultados estão destacados em negrito.

Tabela 28 - Média do número de padrões de corte para as 18 classes.

	Número de padrões de corte								
		Repetição Exaustiva		**Residual**					
	GG	**FFD**	**Gulosa**	**FFD**	**Gulosa**	**Nova 1**	**Nova 2**	**Nova 3**	**A.E.**
C1	4,9	9,45	9,85	4,4	4,85	3,9	3,9	3,8	**2,95**
C2	5,05	11,8	11,55	6,55	6,4	5,15	5,5	5,65	**4,5**
C3	20	32,95	33,25	12,45	13,4	9	9,2	9,1	**7,9**
C4	19,9	40,2	40,2	21,35	21,95	15,85	16,7	17,1	**12,15**
C5	40	61,25	62,05	19,85	21,25	14,45	14,45	14,65	**14,05**
C6	39,8	74,45	73,65	35,35	34,25	27,35	28,25	30,1	**22,85**
C7	10	16,75	17,6	6,75	7,6	5,5	5,7	5,55	**4,6**
C8	9,75	21,4	21,45	12,5	12,4	9,3	10	9,95	**6,9**
C9	20	33,05	34,75	12,35	14,65	8,6	8,6	8,6	**7,75**
C10	19,9	39,55	39,25	21,9	20,9	15,3	15,8	17,3	**12,05**
C11	40	64	69,3	22,9	26,95	14,65	**14,6**	14,7	14,85
C12	39,85	77,55	76,8	40,4	38,5	30	30,4	33,3	**26,4**
C13	10	17,95	19,05	7,95	9,05	5,3	5,25	5,5	**4,35**
C14	9,8	21	21,5	11,9	11,8	8,55	9,4	9,6	**6,75**
C15	20	33,3	33,55	12,85	13,55	8,1	8,05	8,2	**7,9**
C16	19,95	39,2	39,05	20,2	19,35	14,4	14,4	15,95	**11,6**
C17	40	62,65	67,8	21,6	26,7	15,4	15,4	15,4	**14,85**
C18	39,7	78,2	76,15	41,4	39,6	29,6	29,85	33,9	**26,55**
Total	408,6	734,7	746,8	332,65	343,15	240,4	245,45	258,35	**208,95**

O tempo computacional médio (em segundos) para a execução de cada um dos exemplos das 18 classes e o tempo total dos 360 exemplos são mostrados na Tabela 29.

Tabela 29 - Média do tempo computacional paras as 18 classes.

	Tempo computacional (segundos)								
	Construtiva		**Residual**						
	GG	**FFD**	**Gulosa**	**FFD**	**Gulosa**	**Nova 1**	**Nova 2**	**Nova 3**	**A.E.**
cada	10,48222	0	0,142778	26,14556	28,075	18,39	18,56444	18,46444	0,413194
total	3773,6	0	51,4	9412,4	10107	6620,4	6683,2	6647,20	148,75

Os tempos computacionais obtidos na Tabela 29 foram executados em um computador Intel Pentium M (Centrino de 1.73 GHz) com 512 MB de memória RAM.

4.3 Resultados computacionais adicionais

Nesta seção serão apresentados os resultados obtidos com as 18 classes descritas na Tabela 25 utilizando os parâmetros evolutivos da Tabela 26, com exceção da função de avaliação. Outras funções de avaliações, para verificar o comportamento de cada um dos critérios (perda total e números de padrões), forma utilizadas.

As variações da função de avaliação são descritas na Tabela 30.

Tabela 30 - Variações da função de avaliação.

Função	**Parâmetros**	**Descrição**
Função A	$(alfa_1 = 1.0 \text{ e } alfa_2 = 0.0)$	Considera importante somente a perda total do objeto.
Função B	$(alfa_1 = 0.5 \text{ e } alfa_2 = 0.5)$	Considera a mesma importância para ambos os critérios.
Função C	$(alfa_1 = 0.6 \text{e } alfa_2 = 0.4)$	Considera mais importante a perda total do objeto.
Função D	$(alfa_1 = 0.8 \text{ e } alfa_2 = 0.2)$	Considera mais importante a perda total do objeto.
Função E	$(alfa_1 = 0.0 \text{ e } alfa_2 = 1.0)$	Considera importante o número de padrões de corte.

Inicialmente foi avaliada a perda total média dos objetos para cada uma das 18 classes (Tabela 25) considerando as funções de avaliação descritas na Tabela 30 acima.

Quando a função de avaliação considera apenas a perda total (Função A da Tabela 31) a redução do desperdício de objetos chega à aproximadamente a 46,07% em relação à função de avaliação homogênea (função B), porém quando apenas o número de padrões de corte (função E) é considerado o desperdício aumenta em quase 230,60%. Os

resultados obtidos, em função da perda, são apresentados na Tabela 31. Os melhores resultados estão destacados em negrito.

Tabela 31 - Variação da perda em relação a funções de avaliações diferentes.

Classe	Função A	Função B	Função C	Função D	Função E
C1	**66,95**	118,3	107,8	78,25	233,85
C2	**439,5**	520	497,8	461,6	1096,45
C3	**15,9**	39,85	25,65	35,25	91,25
C4	**135,65**	349,5	294	205,6	806,1
C5	**9,7**	70,35	36,75	37,9	107,55
C6	**116**	324,5	318,35	283,55	511,75
C7	**12,9**	58,2	33,1	23,05	108,8
C8	**269,85**	411	390,6	398,5	1214,1
C9	**4,3**	30,8	18,15	16,2	91,85
C10	**97,15**	248,45	245,2	168,9	589
C11	**1,7**	45,1	40,45	46,15	89,9
C12	**124,25**	333,35	301,6	234	655,85
C13	**5,75**	83,25	43,65	34,4	171,2
C14	**64,45**	174,75	162,8	110,7	791,1
C15	**0,75**	15	21	20,5	85,45
C16	**29,9**	104,4	107,15	92,6	217,35
C17	**0,05**	16,7	33,9	17,75	44,55
C18	**131,65**	369,65	340,7	265,8	734,25
Total	**1526,4**	3313,15	3018,65	2530,7	7640,35

A relação da perda considerando funções de avaliações distintas é visualizada no Gráfico 1.

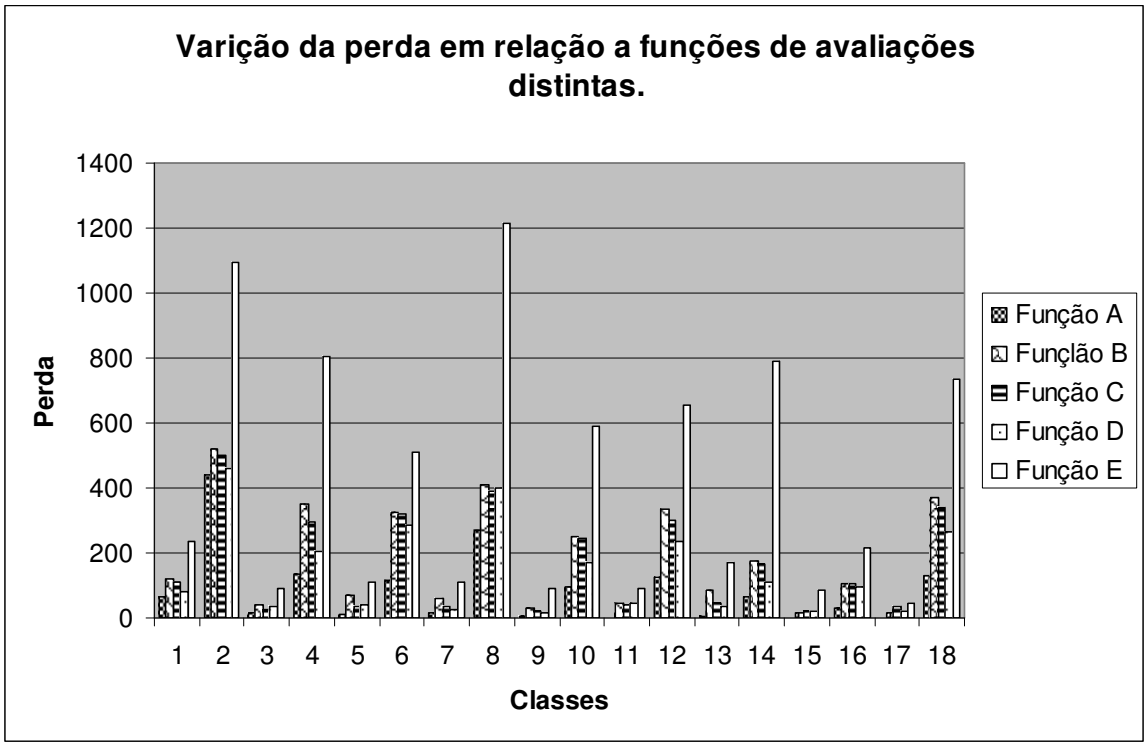


Gráfico 1 - Variação da perda em relação a funções de avaliações distintas.

Uma segunda avaliação, apresentada na Tabela 32, mostra uma variação do número de padrões de corte em relação à funções de avaliações distintas e demonstra que quando é considerado somente o número de padrões de corte (função E da Tabela 30) na função de avaliação, o número total de padrões utilizados é 0,9% a menos em relação a função de avaliação que considera de forma homogênea os dois critérios (função A da Tabela 30). Contudo, quando apenas a perda é considerada, na função de avaliação (função A da Tabela 30), o número de padrões de corte aumenta de forma considerável (aproximadamente 30,66%) em relação à função de avaliação homogênea. Os melhores resultados são destacados em negrito na Tabela 32.

Tabela 32 - Variação do número de padrões em relação a funções de avaliações diferentes.

Classe	Função A	Função B	Função C	Função D	Função E
C1	3,7	2,95	2,95	3,15	**2,75**
C2	5,05	4,5	4,5	4,55	**4,2**
C3	10,65	7,9	8,05	8,4	**7,85**
C4	16,05	**12,15**	12,3	12,3	12,25
C5	19,75	**14,05**	14,2	14,5	14,15
C6	30,45	**22,85**	23,75	23,45	23,25
C7	5,85	**4,6**	**4,6**	4,9	4,65
C8	8,85	**6,9**	7,05	7,05	6,9
C9	10,2	7,75	7,9	7,9	**7,7**
C10	16	12,05	**11,9**	12,2	12
C11	20,95	14,85	15,1	**14,8**	14,95
C12	32,9	**26,4**	26,5	27,2	26,05
C13	5,95	4,35	4,35	4,65	**4,2**
C14	7,75	6,75	6,95	6,9	**6,7**
C15	10,4	7,9	7,95	**7,6**	7,7
C16	15,45	11,6	12,05	**11,35**	11,75
C17	20,15	14,85	**14,7**	15,1	14,95
C18	33,55	**26,55**	26,7	26,8	26,75
Total	273,65	208,95	211,5	212,8	**208,75**

O Gráfico 2 apresenta a relação entre os resultados com as diferentes funções de avaliações (Tabela 32).

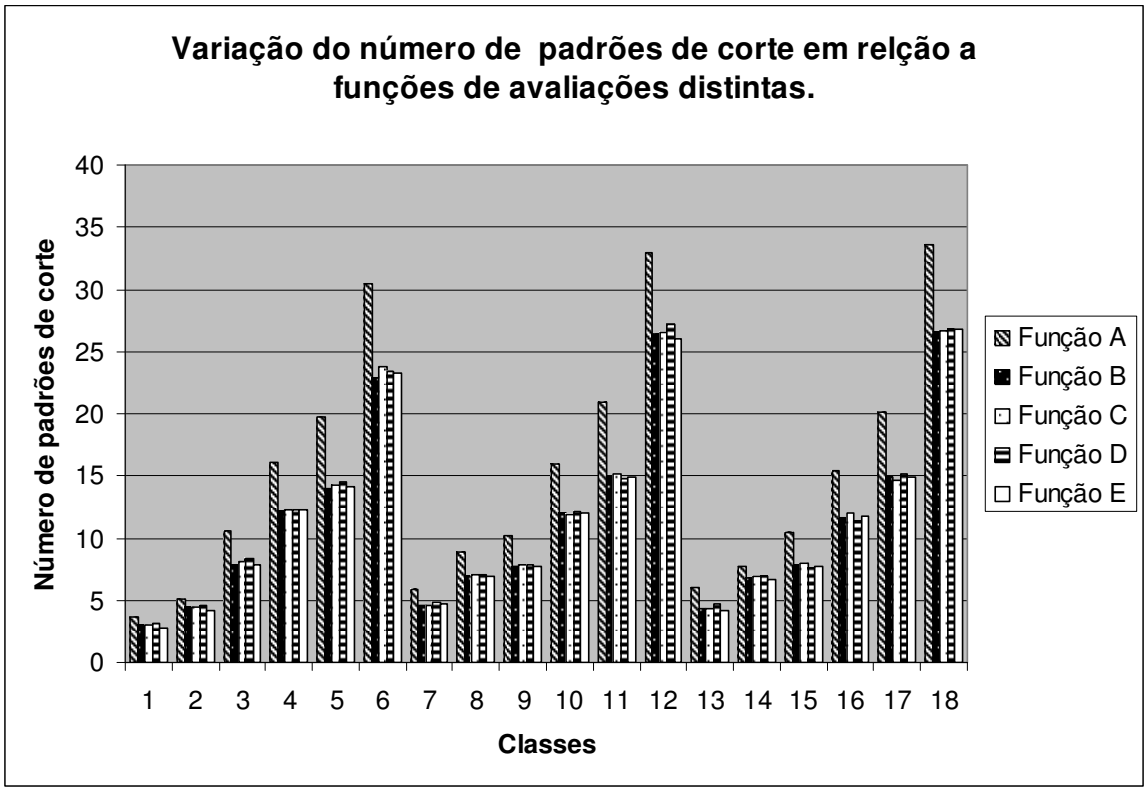


Gráfico 2 - Variação do número de padrões em relação a diferentes funções de avaliações.

O número de objetos cortados foi analisado em relação as funções de avaliações distintas e os seus resultados são visualizados na Tabela 33.

Tabela 33 - Variação do número de objetos cortados em relação a funções de avaliações diferentes.

Classe	Função A	Função B	Função C	Função D	Função E
C1	4,9	4,45	4,9	4,05	**3,65**
C2	**10,8**	12,2	**10,8**	12,15	14
C3	13,8	11,65	13,8	**10,15**	10,4
C4	**32,85**	36,2	**32,85**	35,1	41,4
C5	25,6	**19,15**	25,6	19,55	20,1
C6	52,7	**55,1**	52,7	56,25	57,55
C7	9,2	**6,2**	9,2	6,75	7,15
C8	**21,95**	23,35	21,95	24,35	26,85
C9	15,05	11,2	15,05	11,85	**10,9**
C10	**34,7**	38,85	**34,7**	38,4	43,1
C11	28,05	21,15	28,05	**20,35**	21,7
C12	**73,65**	77,25	**73,65**	78,15	85,1
C13	8,55	7,3	8,55	**7,25**	7,95
C14	**17,75**	19,9	**17,75**	20	22,75
C15	15,1	**10,7**	15,1	10,9	11
C16	**30,65**	32,8	**30,65**	34,95	35,9
C17	31,3	23,7	31,3	**23,35**	24,4
C18	**78,5**	85,45	**78,5**	84,45	89,55
Total	505,1	**496,6**	505,1	498	533,45

As funções que resultaram nos menores números de objetos cortados foram às funções A e C, conforme resultados visualizados em negrito na Tabela 33. E menor número total de objetos cortados foi a função B. O maior número de objetos cortados foi observado na função E que leva em consideração apenas o número de padrões de corte. Os resultados são demonstrados no Gráfico 3.

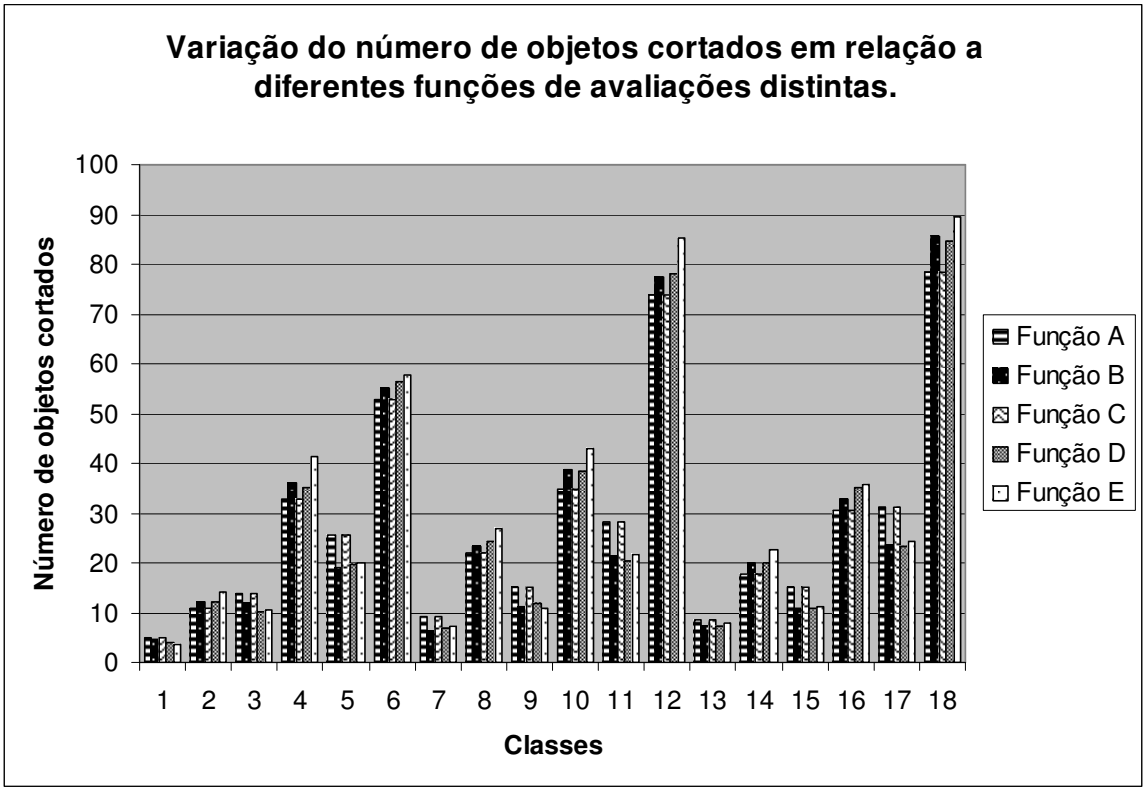


Gráfico 3 –Variação do número de objetos cortados em relação funções de avaliações distintas.

Outros parâmetros além da função de avaliação foram analisados com diferentes variações e os resultados obtidos são descritos a seguir.

O aumento do número de indivíduos na população inicial não expressou grandes melhorias nos resultados. Porém, um aumento no tempo computacional é observado. Cada vez que o tamanho da população é dobrada o tempo computacional varia entre 40% e 80% a mais em relação ao tamanho da população anterior.

Quanto maior o número de genes do indivíduo (parâmetro *g*) a qualidade dos resultados melhora pouco (12% em média) porém um tempo computacional maior é utilizado.

O percentual da escolha dos indivíduos da população para compor o novo indivíduo obteve melhores resultados quando é dada maior probabilidade de seleção aos melhores indivíduos (50%).

Aumentando o número de gerações a qualidade das soluções melhora gradativamente, despendendo um tempo computacional maior.

A diversidade populacional garante uma eficiência maior ou menor do algoritmo evolutivo proposto. A obtenção da população inicial utilizando a mesma probabilidade de geração dos indivíduos através dos métodos propostos (HR2, HR3 e HR4) resultou um

ganho, nos resultados finais, em torno de 25% e 40% superior em relação a qualquer outra probabilidade de geração.

A função de avaliação tem grande influência nos resultados, pode-se dar maior ou menor influência na importância de algum critério da função (perda e número de padrões) e os resultados serão melhores ou piores conforme o critério utilizado.

O tempo computacional médio, em segundos, para a execução de cada um dos exemplos das 18 classes e o tempo total dos 360 exemplos para as funções de avaliações, Tabela 30, são mostrados na Tabela 34.

Tabela 34 - Média do tempo computacional utilizando funções de avaliações distintas.

	Tempo computacional (segundos)				
	Função A	**Função B**	**Função C**	**Função D**	**Função E**
cada	0,437197	0,413194	0,413325	0,447178	0,497569
total	175,391	148,75	148,797	160,9840	179,1250

O gráfico 4 mostra o tempo computacional total para as 18 classes com 20 instancias cada em relação as funções e avaliações distintas apresentadas na Tabela 30.

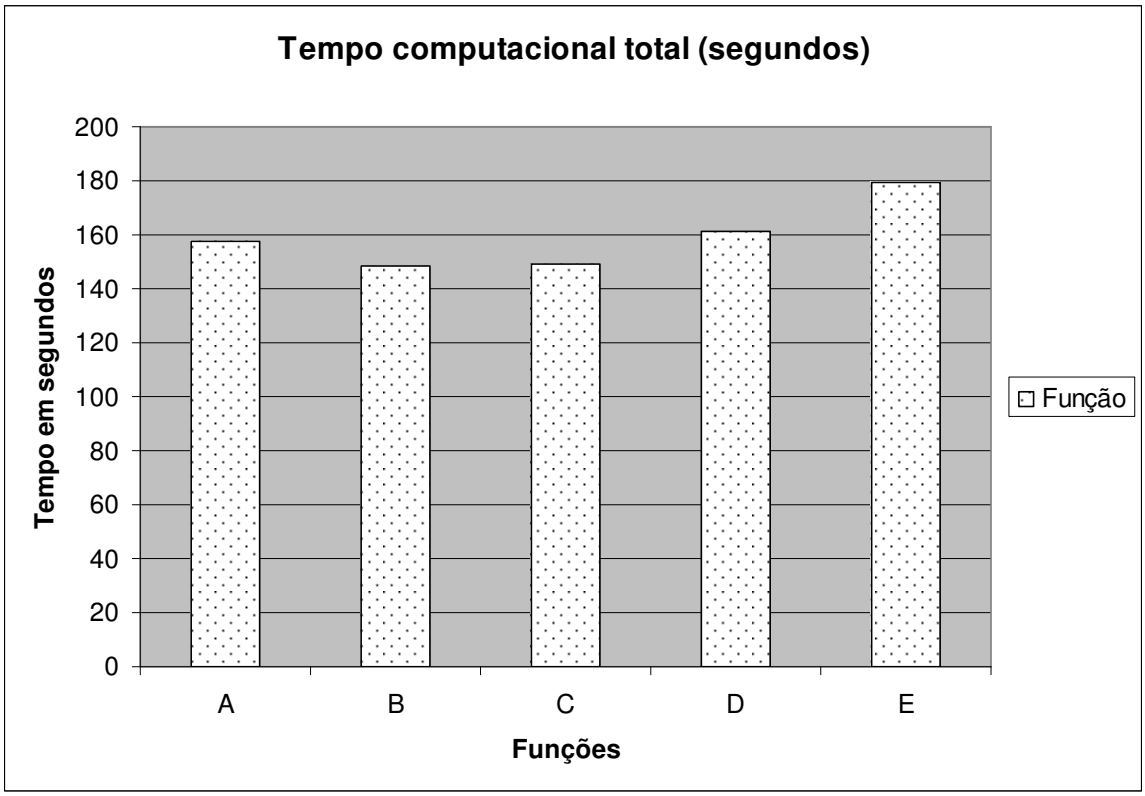


Gráfico 4 - Tempo computacional total (segundos) para os 360 exemplos.

4.3.1 Belov e Scheithauer (2002)

Testes computacionais adicionais, baseados em Belov e Scheithauer (2002) que desenvolveram um algoritmo planar (método exato) para o problema de corte unidimensional com objetos em estoque de tamanhos diferentes, foram realizados. Os exemplos utilizados foram obtidos através do gerador aleatório proposto por Belov e Scheithauer (2002). Esses exemplos estão divididos em 31 classes com 100 exemplos por classe, totalizando 3100 exemplos (instâncias), as características das classes utilizadas são as seguintes:

- número de objetos em estoque: *K=5*;
- número de tipos de itens: *m=40*;
- dimensão dos objetos: L_1*=10.000* e $L_k \in \{5000,5100,...,9900\};\ k = 2,...,5$.
- estoques disponíveis dos objetos $e_k, k = 1,...K$ foram gerados aleatoriamente no intervalo $[2000,2500]$;
- tamanho dos itens $l_i, i = 1,...,m$ foram gerados aleatoriamente no intervalo $[v_1 L_{\max}, v_2 L_{\max}]$, onde $L_{\max} = \max\{L_k\},\ k = 1,...,K;\ v_1 = \{0{,}001,\ 0{,}01,\ 0{,}05,\ 0{,}015,\ 0{,}025\}$ e, $v_2 = \{0{,}2,\ 0{,}3,\ ...,\ 0{,}9\}$
- demanda dos itens $d_i, i = 1,...,m$ são valores no intervalo [200,300].

A função objetivo utilizada por Belov e Scheithauer (2002) é dada pela minimização do custo total do material e do preço dos objetos em estoque que é igual ao tamanho dos objetos em estoque, para eles a função objetivo é equivalente a minimizar o tamanho total dos objetos usados para cortar os itens, isto é, equivalente a minimizar a perda total de material. Para os testes computacionais foi considerada apenas a função de avaliação A (Tabela 30).

A Tabela 35 mostra as características de cada classe gerada (v_1 e v_2) e o resultado computacional (*Gap*). O valor de *Gap* é calculado como:

$$Gap = \frac{Solução\ de\ Belov\ e\ Scheithauer - Solução(A.E.\ função\ A)}{Solução\ de\ Belov\ e\ Shceithauer} x100$$

Tabela 35 - Características das instâncias geradas por Belov e Scheithauer (2002).

Classe	v_1 **e** v_2	***Gap***	**Classe**	v_1 **e** v_2	***Gap***
C1	0,001 e 0,2	0,1755285	**C17**	1,5 e 0,2	1,660874
C2	0,001 e 0,3	0,338222	**C18**	1,5 e 0,3	1,204246
C3	0,001 e 0,4	0,50123	**C19**	1,5 e 0,4	1,157579
C4	0,001 e 0,5	0,785177	**C20**	1,5 e 0,5	1,481326
C5	0,001 e 0,6	0,986865	**C21**	1,5 e 0,6	1,431382
C6	0,001 e 0,7	1,330716	**C22**	1,5 e 0,7	1,684004
C7	0,001 e 0,8	1,843849	**C23**	1,5 e 0,8	2,254709
C8	0,001 e 0,9	1,362868	**C24**	1,5 e 0,9	1,904421
C9	0,001 e 0,2	0,221194	**C25**	2,5 e 0,3	1,213631
C10	0,001 e 0,3	0,366215	**C26**	2,5 e 0,4	2,375498
C11	0,001 e 0,4	0,553312	**C27**	2,5 e 0,5	2,164348
C12	0,001 e 0,5	0,779977	**C28**	2,5 e 0,6	1,535495
C13	0,001 e 0,6	1,044311	**C29**	2,5 e 0,7	2,044762
C14	0,001 e 0,7	1,317463	**C30**	2,5 e 0,8	2,138915
C15	0,001 e 0,8	1,869427	**C31**	2,5 e 0,9	1,553659
C16	0,001 e 0,9	1,263639	**Média**		**1,308529**

Os resultados mostrados na Tabela 35 do A.E. (função A) são muito próximos aos resultados de Belov e Scheithauer (2002), sendo, aproximadamente, 1,30% a média de Gap, mostrando que o método proposto (A.E.) é robusto em relação à variação dos dados. Os resultados obtidos na Tabela 35 foram executados em um computador Intentel Pentim IV (3 GHz) com 512 MB de memória RAM.

5 CONCLUSÕES

O presente trabalho abordou o problema de corte de estoque unidimensional inteiro com objetos em estoque de tamanhos variados e com limite dos objetos em estoque. Foram estudadas abordagens já existentes para a resolução desse problema e também proposta uma nova abordagem. A nova proposta foi desenvolvida utilizando-se os conceitos de Algoritmos Evolutivos. O processo de evolução está dividido em dois sub-processos: cooperação e adaptação. No processo de cooperação uma nova solução é gerada por seleção gulosa/aleatória (com probabilidade maior para os indivíduos com melhor solução) de um indivíduo da população e, depois, a seleção dos genes que farão parte do novo indivíduo de forma aleatória. No processo de avaliação foi verificado se a solução gerada satisfaz ou a não demanda de todos os itens. Caso a demanda de algum item não seja atendida uma heurística de repetição exaustiva utilizando FFD é executada para gerar uma solução factível.

Os padrões de uma solução factível, indivíduo da população, foram obtidos entre 70% a 85% através do A.E. e o restante pela heurística construtiva proposta. Sendo assim o A.E. responsável pela maioria dos padrões de corte de um indivíduo.

O novo método proposto gerou soluções boas em relaç

Em relação a qualidade e o tempo computacional o novo método mostra-se ser eficiente

O novo método proposto parece ser muito promissor, considerando os bons resultados computacionais obtidos em relação às heurísticas comparadas, tendo sido apresentado e discutido em congresso internacional (Araújo et al 2007). O A. E. utilizou-se de uma função multiobjetivo, que leva em consideração os critérios avaliados (perda e número de padrões de corte), os resultados da perda dos objetos foram bons e dos números de objetos melhores ainda. Quando apenas um dos critérios é utilizado na função de avaliação observa-se que este critério possibilitou alcançar resultados melhores para apenas um dos aspectos avaliados (perda e número de padrões de corte).

Algumas extensões trabalho futuros podem ser considerados, como por exemplo: estender o método proposto para problemas bidimensionais; estender o problema para o problema multiperíodo; reduzir o número de diferentes padrões de corte gerados; utilizar outros métodos de geração da população inicial, melhor a qualidade das soluções;

executar o método com outro gerador de dados; comparar os resultados com outros disponíveis na literatura.

REFERÊNCIAS

ALEM. D. J. J.; COSTA, E. F.; ARENALES, M. *O problema de estoque com demanda aleatória: formulação matemática e método de solução*. In: 13 CLAIO – Congresso Latino Iberoamericando de Investigación Operativa. Montevideo, 2006.

ARAUJO, S.; HEIS, A.; CONSTANTINO, A. A. *A genetic algorithm to the one-dimensional cutting sotck problem*. 22 European conference on operation research: books of abstract. Prague, p.44, 2007.

BACK, T. *Evolutionary Algoritms in theory and practice*. Oxford University Press, 1996.

BACK, T.; FOGEL. D. B.; MICHALEWICZ, Z. (eds). *Handbook of evolutionary computation*. Institute of Physics Publishing and Oxford University Press, 1997.

BACK, T.; FOGEL. D. B.; MICHALEWICZ, Z. (eds). *Evolutionary Computation 1: basic algorithms and operators.* Institute of Physics Publishing, 2000a.

BACK, T.; FOGEL. D. B.; MICHALEWICZ, Z. (eds). *Evolutionary Computation 2: advanced algorithms and operators*. Institute of Physics Publishing, 2000b.

BEASLEY, J.E. *Population heuristics*. The management School. Imperial College. London, England. March, 1999.

BELOV, G.; SCHEITHAUER, G. *A cutting plane algorithm for the one-dimensional cutting stock problem with multiple stock lengths*. European Journal of Operational Research, v.141, p.274-294, 2002.

BINKLEY, K. J.; HAGIWARA, M. *Applying self-adaptive evolutionary algorithms to two-dimensional packing problems using a four corner´s heuristic*. Europena Journal of Operacional Research, v.183, p.1230-1248, 2007.

BISCHOFF, E. E.; WÄSCHER, G. (Eds.). *Cutting and Packing*. European Journal of Operation Research, v.84, 1995.

BOLETA, D. A. F.; ARAUJO, S. A.; CONSTANTINO, A. A.; POLDI, K. C. *Uma heurística para o problema de corte de estoque unidimensional inteiro*. In: 37 SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL. Gramado, p.1869-1879, 2005.

CHERRI, A. C.; ARENALES, M. N. *Heurísticas para o problema de corte de estoque com sobras de material reaproveitáveis*. In: 13 CLAIO – Congresso Latino Iberoamericando de Investigación Operativa. Montevideo, 20006.

COSTA, D.; DUBUIS, O. *Embedding a sequential procedure within an evolutionary algorithm for coloring problems in graphs*. Journal of Heuristics, v.1, p.105-128, 1995.

DANTZIG, George B. *Reminiscences about the origins of linear programming*, Oper. Res. Lett. 1 (2) (1981/82), p.43-48.

DEB, K. *Multi-objective optimization using evolutionary algorithms*. John Wiley & Sons, New York, 2001.

DEGRAEVE, M.; PETERS, M. *Optimal integer solution to industrial cutting stock problems: Part 2, Benchmark results, Technical Report*. Katholieke Universiteit Leuven, 2000.

DOWSLAND, K.; DOWSLAND, W. B. *Packing Problems*. European Journal of Operational Research, v.56, p.2-14, 1992.

DYCKHOFF, H.. *A typology of cutting and packing problems.* European Journal of Operational Research, v.44, n.2, p.145-159, 1990.

DYCKHOFF, H..; FINKE, U.. *Cutting and Packing in Production and Distribution – A Typology and Bibliography*. New York, Physica–Verlag, Heidelberg, 258 p., 1992.

DYCHOFF, H.; SCHEITHAUER, G.; TERNO, J.; *Cutting and packing*. In: AMICO, M.; MAFFIOLI, F.; MARTELLO, S. (eds.). Annoted bibliographies in combinatorial optimization. New York, John Wiley & Sons, p.393-414, 1997.

DYCKHOFF, H.; WÄSCHER, G. (Eds.). *Cutting and packing*. European Journal of Operacional Research, v.44, 1990.

EIBEN, A. E.; SMITH, J. E. *Introduction to evolutionary computing*. Springer, 2003.

FARLEY A. A.; RICHARDSON, K. V. *Fixed charge problems with identical fixed charges*. European Journal of Operacional Research, v.18, p.245-249, 1984.

FOERSTER, H.; WÄSCHER, G. *Pattern reduction in one-dimensional cutting stock problem*. International Journal of Production Research, v. 38, p.1657-1676, 2000.

GILMORE, P. C.; GOMORY, R. E. *A linear programming approach to the cutting stock problem*. Operations Research, v.9, p.848-859, 1961.

GILMORE, P. C.; GOMORY, R. E. *A linear programming approach to the cutting stock problem - Part II*. Operations Research, v.11, n.6, p.863-888, 1963.

GILMORE, P. C.; GOMORY, R. E. *Multi-stage cutting stock problems of two and more dimensions*. Operations Research, v.13, p.94-120, 1965.

GILMORE, P. C.; GOMORY, R. E. *The Theory and Computation of Knapsack Functions*. Operations Research, v.14, n.6, p.1045-1074, 1966.

GLOVER, F.; KOCHENBERGER, G. A. *Handbook of metaheuristics*. Kluwer Academic Publishers. Massachusetts, 2003.

GOLDBERG, D. E. *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley, Reading, MA, 1989.

GOULIMIS, C. *Optimal solutions for the cutting stock problem*. Europena Journal of Operational Research, v.44, p.197-208, 1990.

GRADISAR, M.; KLJAJIÉ, M.; RESINOVIC, G.; JESENKO, J. *A sequential heuristic procedure for one-dimensional cuttigng*. European Journal of Operation Research, v.114, p.557-568, 1999.

GRADISAR, M.; RESINOVIC, G.; KLJAJIE, M. *Evaluation of algorithms for one-dimensional cutting*. Computers & Operations Research, v.29, p.1207-1220, 2002.

HAESSLER, R. W. *A heuristic programming solution to a nonlinear cutting stock problem*. Management Scienc, v.17, n.12, p.793-802, 1971.

HAESSLER, R. W. *Controlling cutting pattern changes in one-dimensional trim problems.* Operatios Research, v.23, n.3, p.483-493, 1975.

HAESSLER, R. W.; SWEENEY, P.E. *Cutting stock problem and solution procedures*. European Journal of Operational Research, v54, p.141-150, 1991.

HAUPT, R. L.; HAUPT, S. E.. *Practical genetic algorithms*. 2.ed. Wiley-Interscience publication. New Jersey, 2004.

HERTZ, A.; KOBLER, D. *A framework for the description of evolutionary algoritms*. European Journal of Operation Research, v.126, p.1-12, 2000.

HINXMAN, A. I. *The trim-loss and assortment problems: a survery*. European Journal of Operational Research, v.5, p.8-18, 1980.

HOLLAND, J. H. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control and artificial intelligence*. University of Michigan Press, 1975.

HOLTHAUS, O. *Decomposition approaches for solving the integer one-dimensional cutting stock problem with different types of standard lengths*. European Journal of Operational Research, v.141, p.295-312, 2002.

HOFFMEISTER, F.; BACK, T. *Genetic algorithms and evolution strategies: similarities and difference*. In H.P. Schwefel and R. Manner, editor, Parallel Problem Solving. Springer-Verlag, p.455-469, 1990.

KOZA, J. H. *Genetic Programming: on the programming of computers by means of natural selection*. MIT Press, 1992.

LIEPENS, G.E.; POTTER, W.D. *A genetic algorithm approach to multi-fault diagnosis*. In: Davis, L. (Ed.), Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York, p.237-250, 1991.

MARTELLO, S.; TOTH, P. *Knapsack Problems: algorithms and computer implementations*. New York, J. Wiley & Sons, West Sussex, 1990.

MESSAOUD, S. B.; CHU, C.; ESPINOUSE, M. L. A*n approach to solve cutting stock sheets.* IEEE International Conference on Systems, Man and Cybernetics, p.5109-513, 2004.

MORABITO, R.. *Modelos de otimização para o problema de corte nas indústrias de papel e papelão e de móveis*. Gestão & Produção, v.1, n.1, p.59-76, 1994.

OLIVEIRA, J. F.; WÄSCHER, G. (Eds.). *Cutting and packing*. European Journal of Operacional Research, v.183, 2007.

O´REILLY, U. et al. *Genetic programming theory and practice II*. Springer Science. Boston, 2005.

PILLEGI, G. C. F. *Abordagens para otimização integrada dos problemas de geração e seqüenciamento de padrões de corte*. Tese de Doutorado, ICMC – USP, (2002).

PINTO, M. J. *O problema de corte de estoque inteiro*. Dissertação de mestrado, USP – Universidade de São Paulo, ICMC – Instituo de Ciências Matemáticas e de Computação, São Carlos/SP, 1999.

POLDI, K. C..; ARENALES, M. N.. *Dealing with small demand in integer cutting stock problems with limited different stock lengths*. Tecnhical report ICM-USP, n.85, 2005.

POLDI, K. C.. *Algumas extensões do problema de corte de estoque*. Dissertação de Mestrado, ICMC -USP, 2003.

POLDI, K.; ARENALES, M. N. *O problema de corte de estoque unidimensional multiperíodo*. In: 13 CLAIO – Congresso Latino Iberoamericando de Investigación Operativa. Montevideo, 2006.

RECHEMBERG, I. *Evolution strategie: optimierung techischer systeme nach prinzipien der biologischen evolution*. Frommann-Holzboog, Stuttgart, 1973.

REEVES, C. R. *Modern heuristics techniques for combinatorial problems*. John Wiley & Sons. New York, Toronto, 1993.

REZENDE, S. O. *Sistemas inteligentes: fundamentos e aplicações*. Barueri, SP: Manole, 2005.

SCHEITHAUER, G.; TERNO, J. *The modified integer round-up property of the one-dimensional cutting stock problem*. European Journal of Operational Research, v.84, p.562-571, 1995.

STADTLER, H. A *one-dimensional cutting stock problem in the Aluminium Industry and its solution*. European Journal of Operational Research, v.44, p.209-223, 1990.

SWEENEY, E.; HAESSLER, R. W. *One-dimensional cutting stock decisions for rolls with multiple quality grades*. European Journal of Operation Research, v.44, p. 224-231, 1990.

SWEENEY, P.; PATRNOSTER, E. *Cutting and Packing Problems:* A *Categorized, Application oriented Research Bibliography*. Journal of the Operation Research Society, v.43, p.691-706, 1992.

UMETANI, S.; YAGIURA, M.; IBARAKI, T. One-*dimensional cutting stock problem to minimize the number of different patterns*. European Journal of Operation Research, v.146, p.388-402, 2003.

VANCE, P. *Branch-and-price algoritms for the one-dimensional cuttings stock problem.* Computacional Optimaziton and Apllications, v.9, p.212-228, 1998.

VANDERBECK, F. *Exact algorithm for minimizing the number of setups in the one-dimensional cutting stock problem*. Operations Research, v. 48, p. 915-926, 2000.

WANG, P. Y.; WÄSCHER, G. (Eds.). *Cutting and packing*. European Journal of Operation Research, v.141, 2002.

WÄSCHER, G.; GAU, T. *GUTGEN1: A problem generator for the Standard One-dimensional Cutting Stock Problem*. European Jornal of Operational Research, v.84, p.572-579, 1995.

WÄSCHER, G.; GAU, T. *Heuristics for the integer one-dimensional cutting stock problem: a computational study*. OR Spektrum, v.18, p.131-144, 1996.

WÄSCHER , G.; HAUßNER, H.; SCHUMANN, H. *An Improved Typology of Cutting and Packing Problems*. European Jornal of Operational Research, v.183, p.1109-1130, 2007.

APÊNDICE A – ALGORITMOS EVOLUTIVOS

Algoritmos evolutivos

Neste apêndice serão abordados os algoritmo evolutivos (AE) de uma forma geral. Maiores detalhes podem ser encontrados em Back (1996), Deb (2001), Back et al (1997), Back et al (2000a), Back (2000b), Eiben e Smith (2003), Glover e Kochenberger (2003), Rezende (2005) e Binkley e Hagiwara (2007).

Algoritmos evolutivos são técnicas de otimização baseados no processo de evolução natural e populacional. Muitos autores definem como sendo algoritmo genético. Outros definem como sendo técnicas de solução que consideram uma população de solução.

No processo populacional é descrito uma população inicial de indivíduos (soluções do problema) que evoluem de acordo com algumas regras que são claramente especificadas. Cada iteração (geração de um indivíduo) é alternada com períodos de cooperação e adaptação (Hertz e Kobler, 2000).

Baseado na teoria da evolução natural Back (1996) agrupa os sistemas de computação evolutiva (algoritmos evolutivos) em três grandes categorias:

- algoritmos genéticos: são programas evolutivos baseados na teoria da seleção natural e na hereditariedade. Proposto inicialmente por Holland (1975) se cria no computador uma população de indivíduos representados por seus cromossomos tal como na molécula de DNA do núcleo celular. Os algoritmos genéticos tornaram-se mais populares através do trabalho desenvolvido por Goldberg, 1989. Algumas aplicações de algoritmos genéticos são relatas em Reeves (1993), Beasley (1999), Glover e Kochenberger (2003), Haupt e Haupt (2004), O´Reilly et al (2005), entre outros autores;
- estratégias de evolução: foram propostas por Rechemberg (1973) são utilizadas para resolver problemas hidrodinâmicos e de controle. Estratégias de evolução utilizam mutações normalmente distribuídas para modificar vetores de valores reais. Um comparativo entre estratégias de evolução e algoritmos genéticos podem ser encontradas em Hoffmeister e Back, 1990;

- programação genética: originadas na década de 80 ela procura fornecer um método para a criação automática de programas a partir de uma descrição em alto nível do problema a ser atacado. Consiste em conseguir que computadores façam as tarefas requisitadas a eles sem a necessidade de informá-los como fazê-las. A programação genética procura evoluir uma população de programas de computadores utilizando princípios da genética e evolução natural (Koza, 1992).

O algoritmo básico para métodos baseados em população é dado por Hertz e Kobler (2000) da seguinte forma:

Início do algoritmo

Geração inicial de uma população de indivíduos;
Enquanto o critério de parada não for encontrado *faça*
Cooperação;
Adaptação;
Fim do enquanto

Fim do algoritmo.

Os indivíduos gerados para a população inicial não precisam ser necessariamente soluções factíveis do problema a ser resolvido. Eles podem ser qualquer tipo de solução desde que se tenha um procedimento que possa transformar os indivíduos gerados em soluções viáveis do problema a ser considerado. Após a geração de um novo indivíduo este poderá ou não fazer parte da solução do problema.

A população inicial poderá ser obtida através da geração aleatória de indivíduos obedecendo a condições de contorno previamente estabelecidas pelo usuário ou através uma população inicial fornecida pelo usuário.

Alguns procedimentos são executados para o desenvolvimento dos algoritmos evolutivos e serão detalhados a seguir: processo de seleção, processo de cooperação, processo de adaptação, processo de substituição do indivíduo na população,critério de parada e função de avaliação.

Processo de seleção

O processo de seleção tem por objetivo fazer com que os elementos mais adaptados da geração inicial participem, com maior probabilidade, do processo que irá gerar a nova geração. Vários métodos de seleção tem sido propostos, como por exemplo: o método da roleta, o método do torneio e o método da amostragem universal estocástica.

Método da roleta

O método da roleta é o método mais simples e também o mais utilizado. Os indivíduos de uma geração (ou população) são selecionados para a próxima geração utilizando uma roleta, semelhante à roleta utilizada em jogos de azar. Cada indivíduo da população é representado na roleta por uma fatia proporcional ao seu índice de aptidão. Assim, indivíduos com maiores aptidões ocupam fatias maiores da roleta, enquanto indivíduos de aptidão mais baixa ocupam fatias menores.

Método do torneio

O método de seleção por torneio seleciona *n* indivíduos aleatoriamente, com a mesma probabilidade. O cromossomo com maior aptidão entre estes *n* cromossomos é selecionado para a população intermediária. O processo se repete até que a população intermediária seja preenchida. Geralmente, o valor utilizado para *n* é 3.

Método da amostragem universal estocástica

O método da amostragem universal estocástica é uma variação do método da roleta, sendo que, em vez de uma única agulha, n agulhas igualmente espaçadas são utilizadas, onde n é o número de indivíduos a serem selecionados para a próxima geração. Assim, em vez de n vezes, a roleta é girada uma única vez, exibindo menos variância que as repetidas chamadas do método da roleta.

Processo de cooperação

Durante o período de cooperação as informações dos indivíduos, da população, são trocadas entre si. Cada novo indivíduo é formado por uma associação de indivíduos da população, sendo que cada indivíduo poderá ou não transmitir informações, como por exemplo: sobre seus valores, seus conteúdos, etc.

Quando indivíduos cooperam entre si, para a criação de um novo, apenas um subconjunto destes poderão transmitir suas informações. A geração da população também poderá ser gerada através do histórico da população. A análise das informações dos indivíduos é feita através da memorização dos indivíduos mais adaptados, independentemente deste fazer parte da população atual ou não e uma cooperação entre eles poderá gerar um novo indivíduo.

Os operadores de seleção e de cruzamento (*crossover*) dos algoritmos genéticos podem ser procedimentos de cooperação.

A troca de informação entre os indivíduos poderá resultar em um novo indivíduo infactível durante o processo de cooperação. O tratamento de indivíduos infactíveis será feito no processo de adaptação.

Processo de adaptação

O processo de adaptação consiste no envolvimento dos indivíduos de forma independentemente.

Um indivíduo é considerado factível quando está dentro do espaço de solução. A combinação de informações de um conjunto de indivíduos factíveis poderá resultar em um indivíduo infactível (fora do espaço da solução). Quando indivíduos infactíveis são gerados estes devem ser tratados de forma a resultar indivíduos factíveis.

Liepens e Potter (1991) descrevem três estratégias que podem ser seguidas para que apenas indivíduos factíveis sejam gerados. Uma maneira simples é rejeitar o indivíduo infactível obtido durante o processo de cooperação. Outra abordagem é aceitar um indivíduo infactível e penalizar ele na função de avaliação que mensura a qualidade

de um indivíduo. E a terceira alternativa é desenvolver combinação procedimentos especializados que gerem somente indivíduos factíveis. Está ultima abordagem é freqüentemente usada em procedimento de reparação que transforma indivíduos infactíveis em factíveis

Muitos algoritmos evolutivos podem ser consideravelmente melhorados utilizando algoritmos de busca no processo de adaptação. Algoritmos de busca são procedimentos que tentam melhorar o valor de um indivíduo sem usar informações de outros indivíduos. O uso de um algoritmo de busca tem o propósito de intensificar a busca em algumas regiões do espaço de busca. A técnica de busca local em algoritmos evolutivos tem sido provada ser bem sucedido em muitas aplicações (Costa e Dubuis, 1995). O uso de uma população de indivíduos assegura uma exploração de uma grande parte do espaço de busca, enquanto que os algoritmos de busca ajudam a determinar indivíduos com qualidade alta em regiões identificadas como promissoras.

O sucesso de um algoritmo evolutivo que não usa algum algoritmo de busca pode algumas vezes ser explicado pela transmissão de informações pertinentes durante o processo de cooperação. Sem dúvida, para certos problemas, existe a possibilidade de relatar a qualidade boa de um indivíduo com uma parte particular da informação contida nele. Neste caso, novos indivíduos de qualidade boa podem ser facilmente gerados por partes adequadas mesclada de informações de indivíduos melhores conhecidos.

A convergência prematura do algoritmo em busca de uma solução ótima local é uma das maiores dificuldades encontradas nos algoritmos evolutivos. Para isso estratégias de diversificação são desenvolvidas para que a solução fuja de uma solução ótima local, um procedimento de diversificação pode ser utilizado para modificar aleatoriamente os indivíduos. O procedimento modifica cada indivíduo independentemente, porém o algoritmo de busca pode ter resultados inesperados no sentido que melhorias não sejam obtidas. Podemos citar o procedimento de mutação do algoritmo genético como exemplo deste algoritmo.

Uma outra maneira de mover um indivíduo das regiões de ótimo local é utilizar uma estratégia de diversificação que pode ser aplicada para gerar sistematicamente indivíduos novos em regiões novas como a técnica *Scatter Search*.

O procedimento de mutação dos algoritmos genéticos pode ser parte do processo de adaptação.

Processo de substituição do indivíduo na população

A cada iteração novos indivíduos são gerados e um processo de substituição destes na população dever ser determinado. Existem dois tipos básicos de substituição: substituição geracional e substituição *steady-state*.

Na substituição geracional toda população atual é substituída pela nova população a cada iteração. Na substituição *steady-state* parte da população é modificada.

Critério de parada

Diferentes critérios de parada podem ser utilizados para terminar a execução de um algoritmo evolutivo, por exemplo: após um dado número de gerações (iterações), quando a aptidão média ou do melhor indivíduo não melhorar ou quando as aptidões dos indivíduos de uma população se tornarem muito parecidas. Conhecendo o valor máximo da função objetivo, é possível utilizar este valor como critério de parada.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo