MINISTÉRIO DA EDUCAÇÃO E DO DESPORTO
Escola de Minas da Universidade Federal de Ouro Preto
Departamento de Engenharia de Minas
Programa de Pós-Graduação em Engenharia Mineral - PPGEM

UM ALGORITMO HEURÍSTICO HÍBRIDO PARA MINIMIZAR OS CUSTOS COM A ANTECIPAÇÃO E O ATRASO DA PRODUÇÃO EM AMBIENTES COM JANELAS DE ENTREGA E TEMPOS DE PREPARAÇÃO DEPENDENTES DA SEQUÊNCIA

Autor: Puca Huachi Vaz Penna

Orientador: Marcone Jamilson Freitas Souza

Trabalho de Dissertação apresentado ao Programa de Pós-Graduação em Engenharia Mineral do Departamento de Engenharia de Minas da Escola de Minas da Universidade Federal de Ouro Preto, como parte integrante dos requisitos para a obtenção do título de Mestre em Engenharia Mineral.

Área de concentração: Lavra de Minas.

Ouro Preto, Março de 2009

UM ALGORITMO HEURÍSTICO HÍBRIDO PARA MINIMIZAR OS CUSTOS COM A ANTECIPAÇÃO E O ATRASO DA PRODUÇÃO EM AMBIENTES COM JANELAS DE ENTREGA E TEMPOS DE PREPARAÇÃO DEPENDENTES DA SEQUÊNCIA

Puca Huachi Vaz Penna

Esta dissertação foi apresentada em sessão pública e aprovada em 03 de março de 2009, pela Banca Examinadora composta pelos seguintes membros:

Prof. Dr. Marcone Jamilson Freitas Souza (Orientador - DECOM/UFOP)

Prof. Dr. Carlile Campos Lavor (DMA/UNICAMP)

Prof. Dr. Frederico Gadelha Guimarães (DECOM/UFOP)

Agradecimentos

"A natureza é exatamente simples, se conseguirmos encará-la de modo apropriado. Essa crença tem-me auxiliado, durante toda a minha vida, a não perder as esperanças, quando surgem grandes dificuldades de investigação."
[Albert Einstein]

Gostaria de expressar meus agradecimentos a todos, que de alguma forma, contribuíram para a realização deste trabalho, em especial:

A Deus pelo dom da vida.

Aos meus pais Tattu Penna e Neide Vaz que me ensinaram que o verdadeiro alicerce de nossas vidas é a família.

Às minhas irmãs Uiara, Violeta Camélia, aos irmãos Taiandir e Rairu e aos meus tios, tias, primos e primas, que juntos, me fazem entender e acreditar nos ensinamentos de meus pais.

À Temis, meu amor, que com seu apoio e carinho tem deixado nossas vidas mais felizes.

Ao meu orientador Prof. Marcone pela amizade e ensinamentos, que por sua dedicação a docência e a pesquisa ter despertado em mim esse gosto.

Aos amigos do mestrado, em especial ao Fred, por caminharmos juntos.

Aos professores do Programa de Pós-graduação em Engenharia Mineral da UFOP pelos ensinamentos.

Ao Programa de Pós-graduação em Engenharia Mineral da UFOP pela oportunidade.

À república Quarto Crescente pela amizade.

Resumo

Este trabalho de dissertação tem seu foco no problema de sequenciamento em uma máquina com penalidades por antecipação e atraso da produção. São considerados tempos de preparação da máquina dependentes da sequência de produção, bem como a existência de janelas de entrega distintas. Para resolução do problema, desenvolveu-se um algoritmo heurístico de três fases. A primeira fase baseada em GRASP e Descida em Vizinhança Variável para a geração da solução inicial, a segunda fase baseada em Busca Tabu para refinamento da solução, e por fim, a Reconexão por Caminhos como estratégia de pós-otimização, na terceira fase. Para cada sequência gerada pela heurística é utilizado um algoritmo de tempo polinomial para determinar a data ótima de início de processamento de cada tarefa. Os resultados computacionais mostraram que houve melhoria em relação a um algoritmo da literatura, tanto com relação à qualidade da solução final quanto em relação ao desvio médio.

Palavras-Chaves: Sequenciamento em uma máquina; GRASP; Busca Tabu; Descida em Vizinhança Variável; Reconexão por Caminhos.

Abstract

This work deals with the single machine scheduling problem with earliness and tardiness penalties. Sequence dependent setup times and distinct due windows are considered. To solve this problem, a three-phase heuristic approach was developed. The first phase is based on GRASP and Variable Neighborhood Descent to generate an initial solution; the second phase is based on a Tabu Search for solution refining, and finally Path Relinking is used as a mechanism of post-optimization as a third phase. For each job sequence generated by the heuristic, an optimal timing algorithm is used to determine the completion time for each job in the job sequence. Computational experiments carried out show that previous algorithms found in related literature have been improved, regarding the quality of the final solution and the average gap.

Keywords: *Single Machine Scheduling; GRASP; Tabu Search; Variable Neighborhood Descent; Path Relinking.*

Sumário

1 Preliminares **1**

1.1 Introdução 1

1.2 Justificativa e Relevância 6

1.3 Objetivos 7

1.3.1 Objetivo Geral 7

1.3.2 Objetivos Específicos 7

1.4 Organização do Trabalho 7

2 O Problema de Sequenciamento **9**

2.1 Tipos de Problemas 9

2.2 O Problema de Sequenciamento na Mineração 10

2.3 Problema Abordado 17

3 Revisão Bibliográfica **19**

3.1 Introdução 19

4 Metodologia **26**

4.1 Modelo Matemático 29

4.2 Procedimento de Determinação das Datas Ótimas de Início de Processamento das Tarefas 31

4.3 Modelo Heurístico 38
4.3.1 Representação de uma Solução 38
4.3.2 Vizinhança de uma Solução 38
4.3.3 Função de Avaliação 39
4.3.4 Geração da solução inicial 40
4.3.5 Busca Tabu aplicada ao PSUMAA 43
4.3.6 Reconexão por Caminhos 46
4.3.7 Algoritmo GTSPR 51

5 Resultados Computacionais **52**
5.1 Problemas-teste 52
5.2 Resultados do PLIM 53
5.3 Calibração e Parâmetros do Método Heurístico 54
5.4 Resultados do Algoritmo Heurístico 59

6 Conclusões e Trabalhos Futuros **65**

Referências Bibliográficas **66**

I Resultados Detalhados **71**
I.1 Execução do GTSPR I 71
I.2 Execução do GTSPR II 76

II Publicações **81**

Lista de Tabelas

2.1 Representação de uma solução do problema de alocação de caminhões 12

2.2 Solução de um problema de alocação de caminhões 13

2.3 Tempo de operação dos caminhões 14

2.4 Sequenciamento da Solução da Frente M4 16

2.5 Sequenciamento da Solução da Frente E1 16

2.6 Exemplificação dos dados de entrada 18

5.1 Resultados do PLIM 53

5.2 Parâmetros do Algoritmo GTSPR - Solução Completa 59

5.3 Comparação de resultados de cada fase do algoritmo GTSPR × Gomes Jr. *et al.* (2007) × CPLEX (Parâmetros Principais) 61

5.4 Comparação de resultados de cada fase do algoritmo GTSPR × Gomes Jr. *et al.* (2007) × CPLEX (Parâmetros Secundários) 62

5.5 Comparação dos tempos computacionais GTSPR I × GTSPR II × Gomes Jr. *et al.* (2007) corrigido 64

I.1 Resultados detalhados GTSPR I × Gomes Jr. *et al.* (2007) (8 a 10 tarefas) 72

I.2 Resultados detalhados GTSPR I × Gomes Jr. *et al.* (2007) (11 a 15 tarefas) 73

I.3 Resultados detalhados GTSPR I × Gomes Jr. *et al.* (2007) (20 a 30 tarefas) 74

I.4 Resultados detalhado GTSPR I × Gomes Jr. *et al.* (2007) (40 a 75 tarefas) 75

I.5 Resultados detalhados GTSPR II × Gomes Jr. *et al.* (2007) (8 a 10 tarefas) 77

I.6 Resultados detalhados GTSPR II × Gomes Jr. *et al.* (2007) (11 a 15 tarefas) 78

I.7 Resultados detalhados GTSPR II × Gomes Jr. *et al.* (2007) (20 a 30 tarefas) 79

I.8 Resultados detalhado GTSPR II × Gomes Jr. *et al.* (2007) (40 a 75 tarefas) 80

Lista de Figuras

1.1 Exemplo de sequenciamento sem tempo ocioso entre tarefas 4

1.2 Exemplo de sequenciamento com tempo ocioso entre tarefas 4

2.1 Problema Estudado 18

4.1 Metodologia heurística de solução para o PSUMAA 29

4.2 Programação das tarefas de um bloco na sequência: 1^o Passo 33

4.3 Programação das tarefas de um bloco na sequência: 2^o Passo 34

4.4 Programação das tarefas de um bloco na sequência: 3^o Passo 35

4.5 Programação das tarefas de um bloco na sequência: 4^o Passo 36

4.6 Procedimento PDDOIP 37

4.7 Movimento de troca - N^T 38

4.8 Movimento de realocação - N^R 39

4.9 Movimento de realocação de um bloco - N^{Or} 39

4.10 Procedimento GRASP-VND aplicado ao PSUMAA 40

4.11 Procedimento Constroi_Solucao_GRASP aplicado ao PSUMAA . . . 41

4.12 Procedimento VND_1 aplicado ao PSUMAA 42

4.13 Procedimento Busca Tabu aplicado ao PSUMAA 45

4.14 Procedimento PR aplicado ao PSUMAA 50

4.15 Algoritmo GTSPR 51

5.1 Calibração do parâmetro $GRASPmax$ em função de Fo 54
5.2 Calibração do parâmetro $GRASPmax$ em função do tempo 55
5.3 Calibração do parâmetro $MRDmax$ em função de Fo 55
5.4 Calibração do parâmetro $MRDmax$ em função do tempo 56
5.5 Calibração do parâmetro $TSmax$ em função de Fo 56
5.6 Calibração do parâmetro $TSmax$ em função do tempo 57
5.7 Calibração do parâmetro $|T|$ em função de Fo 57
5.8 Calibração do parâmetro $|T|$ em função do tempo 57
5.9 Calibração do parâmetro $|GE|$ em função de Fo 58
5.10 Calibração do parâmetro $|GE|$ em função do tempo 58
5.11 Calibração do parâmetro $divElite$ em função de Fo 58
5.12 Calibração do parâmetro $divElite$ em função do tempo 59

Lista de Siglas

ACO	*Ant Colony Optimization*
AG	Algoritmos Genéticos
BT	Busca Tabu
FIFO	*First In First Out*
GE	Grupo Elite
GRASP	*Greedy Randomized Adaptive Search Procedures*
ILS	*Iterated Local Search*
LC	Lista de Candidatos
LRC	Lista Restrita de Candidatos
PCV	Problema do Caixeiro Viajante
PDDOIP	Procedimento de Determinação das Datas Ótimas de Início de Processamento das tarefas
PLIM	Programação Linear Inteira Mista
PO	Pesquisa Operacional
PR	*Path Relinking*
PSUM	Problema de sequenciamento em uma máquina
PSUMAA	Problema de sequenciamento em uma máquina com minimização das penalidades por antecipação e atraso
SA	*Simulated Annealing*
VND	*Variable Neighborhood Descent*
VNS	*Variable Neighborhood Search*

Capítulo 1

Preliminares

1.1 Introdução

A Pesquisa Operacional (PO) é a área do conhecimento que usa o método científico para a tomada de decisões, procurando determinar como melhor projetar e operar um sistema, usualmente sob condições que requerem a alocação de recursos escassos (Arenales *et al.*, 2007).

As aplicações em PO são diversas. Dentro do conjunto de problemas estudados encontra-se, com grande destaque, atualmente, a classe de problemas de programação de tarefas (*job scheduling*).

De acordo com França Filho (2007), o estudo de problemas de programação de tarefas envolvendo penalizações pela antecipação e pelo atraso é mais recente que aquele voltado para problemas em que o objetivo envolve uma função não-decrescente do instante de conclusão do processamento da tarefa, tais como tempo médio de fluxo, soma ponderada de atrasos e *makespan* (momento em que termina a execução da última tarefa). Para estes, custos mais elevados decorrem apenas do adiamento da conclusão das tarefas. Entretanto, com a filosofia *just-in-time* adotada por muitas empresas, o foco atual é penalizar também a conclusão das tarefas antes do instante em que elas são requeridas. Isto é justificado pelo fato de que concluir uma tarefa antecipadamente pode resultar em custos financeiros extras pela necessidade antecipada de capital e/ou espaço para armazenamento e/ou de outros recursos para manter e gerenciar o estoque.

Com relação às entregas, são encontrados na literatura três tipos de problemas:

1. os com datas de entrega comuns (*common due date*);
2. os com datas de entrega distintas (*distinct due dates*) e
3. os que têm janelas de entrega distintas (*distinct due windows*).

No primeiro tipo, existe uma única data para entregar todas as tarefas, enquanto no segundo existe uma data de entrega específica associada a cada tarefa. No terceiro tipo, há um período para a conclusão de cada tarefa. Segundo Wan e Yen (2002), esta última situação aparece em situações de incertezas ou tolerâncias com relação às datas de entrega e não há custos se as tarefas forem concluídas dentro da janela de entrega.

Para problemas de produção envolvendo penalizações pela antecipação e pelo atraso com datas comuns de entrega, há muitas propriedades que podem ser consideradas nos algoritmos de solução e que permitem uma redução acentuada do esforço computacional na exploração do espaço de busca. Por exemplo, de acordo com Baker e Scudder (1990), se C_i representa a data de conclusão da tarefa i e $f(C_i)$ o valor da função custo relativo à tarefa i, então na sequência ótima, os pontos $(s_i, f(C_i))$ formam um V (diz-se que a sequência é $V - shaped$). Em outras palavras, na sequência ótima, as tarefas antecipadas devem ser ordenadas de forma não-crescente pela relação entre o tempo de processamento e a penalização pela antecipação, enquanto as tarefas atrasadas devem ser ordenadas de forma não-decrescente pela relação entre o tempo de processamento e a penalização pelo atraso. Outra propriedade interessante é que na sequência ótima não há tempos ociosos entre as tarefas.

Para o caso em que as datas de entrega são distintas, assim como para o caso em há janelas de entrega distintas, tais propriedades não são necessariamente válidas, tornando mais complexa a exploração do espaço de busca. Para esses tipos de problemas é necessário saber se é permitida a ociosidade das máquinas ou não.

Segundo França Filho (2007), existem situações em que a ociosidade não é permitida por gerar custos maiores que aqueles decorrentes da conclusão antecipada das tarefas. Entretanto, há situações em que vale a pena manter uma máquina ociosa, mesmo que haja uma tarefa em condições de ter seu processamento iniciado. Para ilustrar esta situação, sejam duas tarefas i e j, que devem ser consecutivas em uma sequência de produção envolvendo uma máquina, às quais estão associados os custos unitários por atraso de 100 e 20, respectivamente, custos unitários por antecipação de 22 e 2, e que tenham como datas de entrega os instantes $T_i = 50$ e $T_j = 55$, respectivamente, e 15 e 25 unidades de tempo de processamento, respectivamente. Suponha que a máquina já esteja disponível desde o instante 20. Se a tarefa i começar no instante 20, será concluída no instante 35 e, portanto, haverá uma penalização igual a 330 ($= 15 \times 22$) pela antecipação. Nesse caso, a tarefa j poderá começar somente no instante 35. Considerando seu tempo de processamento, ela será concluída no instante 60, portanto 5 unidades de tempo atrasada em relação à data de entrega, resultando em um custo por atraso igual a 100 ($= 5 \times 20$), e um custo total de 430 ($= 330 + 100$), conforme pode ser visto na figura 1.1. Considere, agora, que o processamento da tarefa i seja iniciado no instante 35, mesmo sabendo que a máquina está disponível desde o instante 20. Neste caso, a tarefa i seria concluída no instante 50, sem multa nem por antecipação nem por atraso. A tarefa j, então, já poderia começar, terminando no instante 75, com 20 unidades de tempo de atraso, resultando em uma multa por atraso de 400 ($= 20 \times 20$), mostrado na figura 1.2. O custo total nesta última situação seria de 400 unidades, menor que aquele no qual a tarefa i começava desde o instante de liberação da máquina. Assim, não havendo restrições à ociosidade das máquinas, determinar a melhor data para iniciar o processamento de cada tarefa ou, equivalentemente, inserir tempos ociosos entre tarefas, poderá produzir soluções de menores custos.

Com relação à influência do tempo de preparação das tarefas na sequência de produção, chamado de tempo de *setup*, observa-se que a maioria dos trabalhos científicos considera que tais tempos são negligenciáveis ou, então, podem ser adicionados aos tempos de processamento das tarefas. Isto é, considera-se que os tempos de *setup* são independentes da sequência de tarefas. Contudo, muitos estudos, como

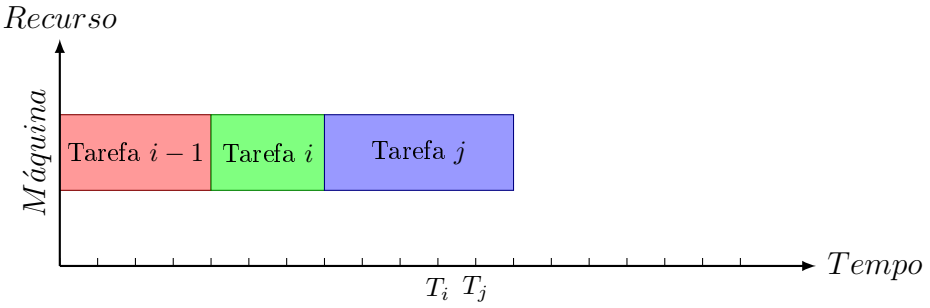

Figura 1.1: Exemplo de sequenciamento sem tempo ocioso entre tarefas

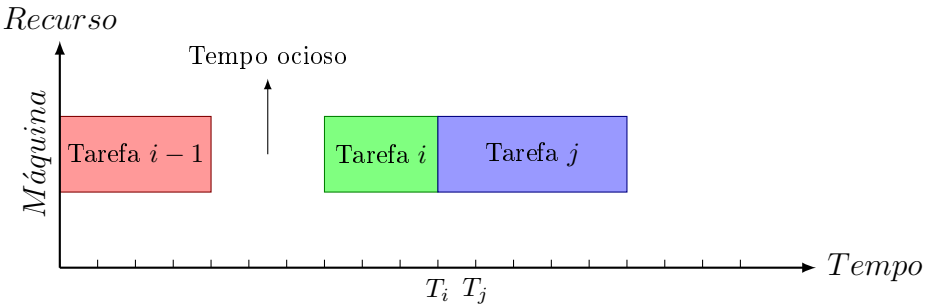

Figura 1.2: Exemplo de sequenciamento com tempo ocioso entre tarefas

os de Panwalkar *et al.* (1973), *apud* Gupta e Smith (2006), indicam que tempos de *setup* significativos aparecerem com frequência em muitas situações práticas sempre que há alteração da tarefa a ser processada na máquina. Segundo os autores, com base em informações prestadas por gerentes de produção, cerca de 70% dos entrevistados afirmaram que em pelo menos 25% das operações, o tempo de preparação era dependente da sequência de produção. De acordo com Kim e Bobrowski (1994), para modelar melhor casos práticos, esses tempos devem ser considerados explicitamente sempre que eles forem significativamente maiores que os tempos de processamento. Uma revisão de trabalhos sobre sequenciamento da produção com tempos de preparação pode ser encontrada em Allahverdi *et al.* (1999).

Ainda segundo Allahverdi *et al.* (1999), os problemas de sequenciamento envolvendo tempo de preparação podem ser classificados em dois tipos de problemas:

1. com tempo de preparação independente da sequência (*sequence-independent*);
2. com tempo de preparação dependente da sequência (*sequence-dependent*).

No primeiro tipo de problema o tempo de preparação depende somente da tarefa a ser processada, já no segundo, o tempo de preparação depende da tarefa a ser processada e da tarefa que a imediatamente precede. Neste caso, de acordo com Allahverdi *et al.* (1999), alguns autores resolveram este problema através do Problema do Caixeiro Viajante - PCV (*traveling salesman problem* - TSP), em que os tempos de preparação são tratados como distâncias entre as cidades.

Dentre os problemas de produção, o de sequenciamento em uma máquina com minimização das penalidades por antecipação e atraso da produção (*Single Machine Scheduling for Minimizing Earliness and Tardiness Penalties*), representado simplesmente por PSUMAA, é um dos mais simples.

Além da importância prática desse problema, com inúmeras aplicações industriais (Allahverdi *et al.*, 1999), trata-se de um problema difícil de ser resolvido na otimalidade em tempos computacionais aceitáveis, por pertencer à classe NP-difícil (Du e Leung, 1990; Liaw, 1999; Biskup e Feldmann, 2001; Wan e Yen, 2002). Essa conjunção de aplicabilidade e dificuldade de resolução na otimalidade tem motivado os pesquisadores a desenvolverem algoritmos eficientes para resolvê-lo.

Neste trabalho, trata-se o PSUMAA com janelas de entrega distintas e tempos de preparação da máquina dependentes da sequência da produção. Surpreendentemente, essa variante do PSUMAA ainda não tem merecido a devida atenção. O único trabalho encontrado na literatura, com essas características, é o de Gomes Jr. *et al.* (2007).

Para resolvê-lo, propõe-se um algoritmo de 3 fases, combinando os procedimentos GRASP (Feo e Resende, 1995), *Variable Neighborhood Descent* - VND (Mladenović e Hansen, 1997), Busca Tabu (Glover, 1986) e Reconexão por Caminhos (*Path Relinking*) (Glover, 1996). Na primeira fase é utilizado um procedimento baseado em GRASP para gerar uma solução inicial. Nesta fase, cada solução construída é refinada por um procedimento baseado em Descida em Vizinhança Variável (VND), que utiliza movimentos de realocação e troca de tarefas para explorar o espaço de soluções. Na segunda fase, a solução proveniente do GRASP é refinada por um procedimento Busca Tabu. Durante essa fase é formada uma lista de soluções

elite (grupo elite), contendo soluções de alta qualidade e diferenciadas entre si. Como pós-otimização, na terceira e última fase, é aplicada a Reconexão por Caminhos a pares de soluções do grupo elite.

1.2 Justificativa e Relevância

O problema de sequenciamento da produção está entre os mais estudados desta década. Segundo Allahverdi *et al.* (2008) em seu trabalho de revisão bibliográfica sobre o problema de sequenciamento com tempo de preparação, entre os anos de 1999 e 2005, foram publicados em periódicos mais de 300 trabalhos sobre o assunto, uma média superior a 40 trabalhos por ano. Este número é muito superior aos 190 trabalhos encontrados entre as décadas de 60 e 90.

Apesar do grande número de trabalhos publicados sobre o problema de sequenciamento, na revisão de Allahverdi *et al.* (2008) não foram encontrados trabalhos sobre o PSUMAA com janelas de entrega distintas e tempos de preparação da máquina dependentes da sequência da produção, foco deste trabalho.

De acordo ainda com Allahverdi *et al.* (2008), a metaheurística Busca Tabu é uma das mais utilizadas para resolver esta classe de problemas. Como esta técnica requer uma solução inicial de qualidade, nada mais natural que utilizar a metaheurística GRASP para gerá-la. Por sua vez, somente a fase de construção GRASP não guia a ótimos locais com relação às estruturas de vizinhanças adotadas. Assim, aplica-se também um método de busca local para melhorar as soluções construídas. A metaheurística VND foi usada para esse fim, pois requer poucos parâmetros (basicamente a definição das vizinhanças e a ordem de exploração destas), produz soluções de boa qualidade (Mladenović e Hansen, 1997) e é de fácil implementação. A aplicação de Reconexão por Caminhos deveu-se ao sucesso desta técnica na solução de vários outros problemas combinatórios (Glover e Kochenberger, 2003).

Este trabalho apresenta relevância por tratar um subproblema de sequenciamento ainda pouco estudado e contribuir com o desenvolvimento de uma ferramenta computacional para a otimização de processos produtivos nas indústrias têxtil, pe-

troquímica, siderúrgica e mineração, dentre outras.

1.3 Objetivos

1.3.1 Objetivo Geral

Este trabalho possui como objetivo geral desenvolver um algoritmo eficiente de otimização, baseado em técnicas metaheurísticas, para resolver o problema de sequenciamento em uma máquina com minimização das penalidades por antecipação e atraso da produção, com tempo de preparação dependente da sequência de produção.

1.3.2 Objetivos Específicos

Os objetivos específicos a serem atingidos são os seguintes:

1. Fazer uma revisão de literatura sobre as metodologias utilizadas para resolver o problema de sequenciamento em uma máquina;
2. Fazer uma revisão de literatura sobre as técnicas metaheurísticas GRASP, *Variable Neighborhood Descent* - VND, Busca Tabu e Reconexão por Caminhos (*Path Relinking*);
3. Desenvolver uma metodologia de otimização híbrida, baseada em metaheurísticas, para resolver eficientemente o problema;
4. Contribuir com a formação de recursos humanos especializados nessa área do conhecimento;
5. Possibilitar maior eficiência no planejamento do sequenciamento da produção.

1.4 Organização do Trabalho

O restante deste trabalho está estruturado como segue.

No capítulo 2, as características do problema estudado são apresentadas em detalhes, bem como uma possível aplicação na área de Mineração.

Uma revisão dos trabalhos correlatos e das técnicas heurísticas utilizadas são apresentadas no capítulo 3.

No capítulo 4 é descrito o algoritmo desenvolvido para resolução do PSUMAA.

No capítulo 5 são apresentados e discutidos os resultados computacionais.

Por fim, o capítulo 6 conclui o trabalho e aponta perspectivas futuras para melhoramento do algoritmo proposto.

Capítulo 2

O Problema de Sequenciamento

Neste capítulo é apresentado os principais tipos de problemas de sequenciamento da produção e, em detalhes, o problema estudado neste trabalho.

2.1 Tipos de Problemas

Na revisão de Allahverdi *et al.* (2008), os autores apresentam os principais tipos de problema de sequenciamento, são eles:

1. **Uma máquina (*single-machine*)**: Neste tipo de problema, foco deste trabalho, todas as tarefas são processadas em uma única máquina;

2. **Máquinas paralelas (*parallel machines*)**: Existem m máquinas em paralelo, que podem ser idênticas ou não. Cada tarefa pode ser processada em qualquer uma das máquinas;

3. ***Flow shop***: Em um ambiente *flow shop* com m máquinas há m estágios em série, com uma ou mais máquinas em cada estágio. Cada tarefa tem que ser processada em cada um dos m estágios na mesma ordem. O tempo de processamento para cada tarefa em diferentes estágios podem ser diferentes. Os problemas *flow shop* podem ser classificados como:

   (a) ***regular flow shop***: Existe uma máquina em cada estágio;

(b) ***no-wait flow shop***: Uma operação sucessora inicia imediatamente após sua antecessora terminar;

(c) ***flexible (hybrid) flow shop***: mais de uma máquina existe em pelo menos um estágio;

(d) ***assembly flow shop***: Em um ambiente *assembly flow shop* de dois estágios, cada tarefa passa por $m-1$ operações específicas em uma máquina predeterminada no primeiro estágio, seguida de uma operação de montagem (*assembly operation*) na máquina do segundo estágio;

4. ***Job shop***: Consiste em m máquinas diferentes e cada tarefa possui uma dada ordem de execução, na qual algumas máquinas podem não ser usadas e outras podem ser repetidas;

5. ***Open shop***: Este ambiente é semelhante ao anterior, porém cada tarefa deve ser processada uma única vez em cada uma das m máquinas, passando por elas em qualquer ordem.

2.2 O Problema de Sequenciamento na Mineração

A mineração realiza suas atividades em minas subterrâneas ou a céu aberto. Em uma mina a céu aberto, o planejamento operacional de lavra consiste no planejamento de curto prazo, onde o principal objetivo é a determinação do ritmo de lavra a ser implementado em cada frente, fornecendo à usina de beneficiamento uma alimentação adequada.

Vários trabalhos tratam esse problema em suas várias etapas, entre elas o problema de mistura de minérios, alocação de equipamentos de carga e transporte e o problema de despacho, de forma independente ou integrada.

Segundo Gershon (1982), *apud* Costa (2005), otimizar o problema de planejamento de lavra em partes independentes pode gerar conflitos que inviabilizam a implementação das soluções obtidas em cada parte.

Costa (2005) estudou três problemas do planejamento operacional de lavra,

a mistura de minérios, mistura de minérios com alocação dinâmica de caminhões e mistura de minérios com alocação estática de caminhões. Os problemas foram resolvidos através de duas metodologias, uma baseada em programação matemática e outra utilizando técnicas heurísticas.

Rodrigues (2006) fez uma análise comparativa das metodologias de despacho de caminhões em minas a céu aberto. Para resolver o problema foi utilizado programação linear, programação dinâmica e uma heurística baseada em Algoritmos Genéticos (AG).

Araújo (2008) tratou o planejamento operacional de lavra com alocação dinâmica de caminhões. O problema abordado consistia em determinar o número de viagens que cada caminhão devia fazer a cada frente de lavra, bem como decidir em quais frentes de lavra alocar as carregadeiras, com a finalidade de atender às metas de produção e qualidade requeridas para o minério a ser produzido, fazendo o melhor aproveitamento da frota de veículos disponível. Na resolução do problema foi empregado um procedimento heurístico baseado na metaheurística *Iterated Local Search*, e um modelo matemático foi formulado para validar o método heurístico proposto.

A presente pesquisa tem seu foco no sequenciamento da produção, que é uma etapa do planejamento operacional de lavra que surge após a alocação dos equipamentos de carga (carregadeiras) e transporte (caminhões) às frentes.

Costa (2005) descreve que o problema da mistura de minérios com alocação de caminhões objetiva determinar o ritmo de lavra de cada frente de acordo com sua capacidade de produção. A capacidade de produção de cada frente é determinada pelos equipamentos de carga nela alocada e pelos caminhões que realizam o transporte do material até o ponto de basculamento. Neste problema há operação de lavra de minério e estéril, a mina conta com uma frota limitada de equipamentos de carga, os quais devem ser alocados às frentes de lavra e operar em uma faixa de produtividade que torne viável sua utilização. O transporte do material retirado da frente de lavra é realizado por uma frota de caminhões com capacidade de carga diferentes.

Esses caminhões são alocados às frentes de lavra, tentando-se evitar a formação de filas, ou seja, o caminhão é alocado a um ponto de carga ou basculamento que proporcione o menor tempo de fila possível. Entretanto, um caminhão somente pode ser direcionado à uma frente que possua um equipamento de carga compatível.

Para exemplificá-lo, seja a Tabela 2.1, a qual apresenta a solução do problema de alocação dinâmica de caminhões de Araújo (2008). Nesta tabela, a primeira coluna indica as frentes de lavra; a segunda, as carregadeiras e o *status* da alocação (1 se ela está em operação e 0, caso está inativa) e as demais colunas, os caminhões. Cada célula desta tabela indica o número de viagens que cada caminhão faz à frente de lavra. Um número negativo indica incompatibilidade entre o caminhão e a carregadeira alocada à frente. Pode-se observar que a carregadeira 5 está alocada e em operação na frente de lavra F_1. Na frente F_2 a carregadeira 3 está alocada, porém não está em operação. Já na F_3 não há nenhuma carregadeira alocada. Em relação aos caminhões, a tabela indica que o Cam_1 efetua 5 viagens à frente F_1, não é compatível com a carregadeira da frente F_2, não faz viagens a frente F_3, faz 7 viagens a frente F_4, e assim por diante.

Tabela 2.1: Representação de uma solução do problema de alocação de caminhões

Frentes	*Carga*	Cam_1	Cam_2	$\cdots$	$Cam_{\|V\|}$
F_1	(5, 1)	5	0	$\cdots$	2
F_2	(3, 0)	-1	-1	$\cdots$	0
F_3	(-1, 0)	0	0	$\cdots$	-1
F_4	(6, 1)	7	-1	$\cdots$	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$F_{\|F\|}$	(2, 1)	4	0	$\cdots$	1

Para mostrar como o problema de sequenciamento aparece no planejamento operacional de lavra, suponhamos a solução do problema de alocação de caminhões, dada na Tabela 2.2. Nesta tabela são descritos o número de viagens alocadas para cada caminhão, C_0 à C_{27}, às frentes de lavra, sendo as linhas M_1 a M_{12}, frentes de minério e E_1 à E_5 frentes de estéril.

Tabela 2.2: Solução de um problema de alocação de caminhões

Fr.	C_{01}	C_{02}	C_{03}	C_{04}	C_{05}	C_{06}	C_{07}	C_{08}	C_{09}	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}	C_{16}	C_{17}	C_{18}	C_{19}	C_{20}	C_{21}	C_{22}	C_{23}	C_{24}	C_{25}	C_{26}	C_{27}
M_1																											
M_2																3				5							5
M_3																		3	5					4			
M_4		2		2			5		4	4																	
M_5																											
M_6																											
M_7																											
M_8	1					5					5	4			3												
M_9																					5	1	5	1			
M_{10}																											
M_{11}																											
M_{12}																2	5								5	1	
E_1		3		3	5			2		1		1	1		2												
E_2																											
E_3																											
E_4	4		5										4	5													
E_5																											

A Tabela 2.3 mostra os tempos de operação dos caminhões em cada frente de lavra, onde as linhas representam as frentes de lavra, e as colunas o tempo de deslocamento vazio, o tempo de deslocamento cheio, o tempo de carga, o tempo de descarga e o tempo total, respectivamente.

Tabela 2.3: Tempo de operação dos caminhões

Frentes	**TDVazio**	**TDCheio**	**Tcarga**	**Tdescar**	**Ttotal**
M_1	2	3	3	1	9 min
M_2	3	4	3	1	11 min
M_3	3	4	3	1	11 min
M_4	3	4	3	1	11 min
M_5	4	6	3	1	14 min
M_6	2	3	3	1	9 min
M_7	2	4	3	1	10 min
M_8	3	4	3	1	11 min
M_9	2	3	3	1	9 min
M_{10}	4	5	3	1	13 min
M_{11}	2	4	3	1	10 min
M_{12}	3	4	3	1	11 min
E_1	3	5	3	1	12 min
E_2	2	4	3	1	10 min
E_3	2	3	3	1	9 min
E_4	3	4	3	1	11 min
E_5	4	6	3	1	14 min

As Tabelas 2.4 e 2.5 representam uma possibilidade de sequenciamento nas frentes M_4 e E_1, respectivamente, para o problema apresentado na Tabela 2.2, em um intervalo de uma hora de produção.

Nas Tabelas 2.4 e 2.5, as colunas indicam os minutos e as linhas a operação de cada caminhão. O caracter c indica que o caminhão está sendo carregado naquele instante; o caracter d, que o caminhão está basculando (descarregando o material) e uma célula vazia indica que o caminhão está em viagem (cheio ou vazio) ou aguardando disponibilidade da carregadeira para carregamento. O caracter o indica a existência de tempo ocioso da carregadeira. Esse último caracter aparece somente na última linha, não vinculado a nenhum caminhão, apenas para mostrar a ociosidade da carregadeira. Neste exemplo, considera-se que os caminhões possuem a mesma capacidade e são compatíveis com as carregadeiras alocadas a cada frente.

Os tempos adotados foram aqueles apresentados na Tabela 2.3.

Nesta situação, pode-se considerar uma tarefa como o carregamento de um caminhão e uma máquina como sendo a carregadeira alocada à frente. Assim, tem-se que a carregadeira alocada à frente M_4, dita máquina M_4, executa 17 tarefas, com tempo total de processamento de 59 minutos. Uma solução para o seu sequenciamento pode ser dado pelo vetor $v_{M_4} = \{C_{02}, C_{04}, C_{07}, C_{02}, C_{04}, C_{07}, C_{09}, C_{10}, C_{07}, C_{09}, C_{10}, C_{07}, C_{09}, C_{10}, C_{07}, C_{09}, C_{10}\}$. Já a máquina E_1 executa 18 tarefas, com tempo total de processamento de 55 minutos, cujo vetor solução pode ser $v_{E_1} = \{C_{05}, C_{08}, C_{10}, C_{12}, C_{05}, C_{08}, C_{12}, C_{02}, C_{04}, C_{05}, C_{15}, C_{02}, C_{04}, C_{05}, C_{15}, C_{02}, C_{04}, C_{05}\}$.

É possível observar que ao tratar as carregadeiras alocadas às frentes de lavra como máquinas, caracteriza-se a existência de várias máquinas; portanto, o problema é classificado como sequenciamento em máquinas paralelas. A solução deste problema envolve, assim, o sequenciamento em cada máquina, foco deste estudo. Vale ressaltar que este trabalho trata o problema de sequenciamento de forma bem mais ampla e genérica, sendo possível ajustar seus parâmetros e/ou acrescentar outros critérios otimizantes para que se comporte como descrito acima.

Ao tratar o sequenciamento da produção, no intervalo de apenas uma hora de produção, é possível identificar características do PSUM, como a máquina, a tarefa, o tempo de processamento e a ociosidade entre as tarefas. Porém, ao se ampliar o horizonte de planejamento, pensando em termos semanais, por exemplo, é possível também identificar outras características como tempo de *setup*, que pode ser visto como o tempo necessário para uma carregadeira se deslocar de uma frente para outra. Ao se estabelecer a produção de materiais diferentes e um período de entrega distinto para cada um desses materiais, pode-se ter como objetivo entregar os produtos nos períodos indicados, minimizando-se os custos com o atraso e/ou com a antecipação da produção. O objetivo de minimizar os custos com a antecipação pode ser devido ao custo de estocagem do material até sua entrega, enquanto que o objetivo de minimizar os custos com o atraso visa à redução das multas contratuais por não entregar o produto na data requerida.

O problema tratado neste trabalho é apresentado a seguir na seção 2.3.

Tabela 2.4: Sequenciamento da Solução da Frente M4

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**
C_{02}	c	c	c					d				c	c	c					d																																										
C_{04}				c	c	c					d				c	c	c					d																																								
C_{07}							c	c	c					d				c	c	c					d				c	c	c					d				c	c	c					d				c	c	c				d				
C_{09}																					c	c	c				d				c	c	c					d				c	c	c					d				c	c	c						
C_{10}																								c	c	c					d				c	c	c					d				c	c	c				d				c	c	c		
										o	o																o	o											o	o									o	o										

Tabela 2.5: Sequenciamento da Solução da Frente E1

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**
C_{02}																							c	c	c						d				c	c	c				d				c	c	c								d						
C_{04}																										c	c	c						d				c	c	c				d				c	c	c				d				
C_{05}	c	c	c					d					c	c	c						d								c	c	c						d				c	c	c				d				c	c	c						
C_{08}				c	c	c						d				c	c	c						d																																					
C_{10}							c	c	c						d																																																
C_{12}										c	c	c						d																																												
C_{13}																			c	c	c						d																																				
C_{15}																																c	c	c						d				c	c	c						d								
																						o																																											

2.3 Problema Abordado

O problema de sequenciamento em uma máquina com penalidades por antecipação e atraso (PSUMAA) tratado neste trabalho tem as seguintes características:

(i) Uma máquina deve processar um conjunto de n tarefas;

(ii) A cada tarefa i está associado um tempo de processamento P_i e uma janela de entrega $[E_i, T_i]$, indicando uma data inicial E_i e uma data final T_i desejadas para o término de seu processamento. Se a tarefa i for finalizada antes de E_i há um custo α_i por unidade de tempo de antecipação. Caso a tarefa seja finalizada após T_i então há um custo β_i por unidade de tempo de atraso. As tarefas concluídas dentro da janela de entrega não proporcionam nenhum custo;

(iii) A máquina pode executar no máximo uma tarefa por vez e uma vez iniciado o processamento de uma tarefa, não é permitida a sua interrupção;

(iv) Todas as tarefas estão disponíveis para processamento na data 0;

(v) Entre duas tarefas i e j consecutivas é necessário um tempo S_{ij} de preparação da máquina, chamado tempo de setup. Assume-se que a máquina não necessita de tempo de preparação inicial, ou seja, o tempo de preparação da máquina para o processamento da primeira tarefa na sequência é igual a 0;

(vi) É permitido tempo ocioso entre a execução de duas tarefas consecutivas.

A Figura 2.1 exemplifica o PSUMAA com janelas de entrega distintas e tempos de preparação da máquina dependentes da sequência da produção, foco deste trabalho.

Segundo a notação $a|b|c$ de Allahverdi *et al.* (2008), o primeiro campo, a, indica o tipo de máquina; o segundo campo, b, indica detalhes das características de processamento, como tempo de preparação e datas de entrega e, por fim, o terceiro campo, c, indica o objetivo a ser minimizado. Desta forma, podemos descrever o PSUMAA com a seguinte notação $1|ST_{sd}, E_i \leq d_i \leq T_i| \sum_{i=1}^{n}(\alpha_i e_i + \beta_i t_i)$, onde:

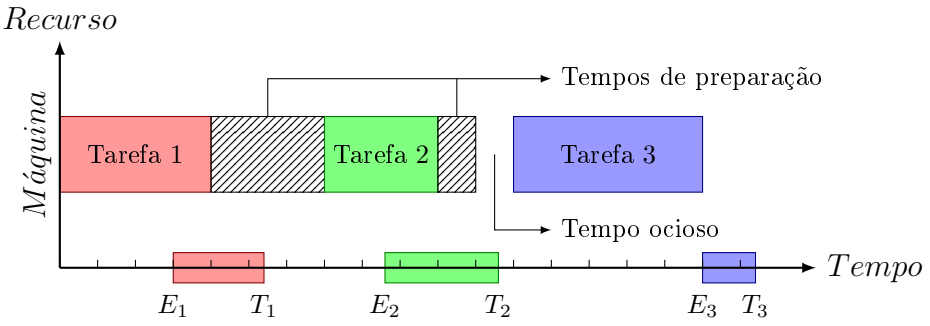

Figura 2.1: Problema Estudado

1. Tipo de máquina (Campo a):
   (a) 1, indicando uma máquina (*single machine*);
2. Característica de processamento (Campo b):
   (a) ST_{sd}, representando tempo de preparação dependente da sequência (*sequence-dependent setup time*);
   (b) $E_i \leq d_i \leq T_i$, indicando janela de entrega distinta (*distinct due window*);
3. Função objetivo (Campo c):
   (a) $\sum_{i=1}^{n}(\alpha_i e_i + \beta_i t_i)$, representando penalização por atraso e antecipação (*total weighted earliness and tardiness*).

Para exemplificar o problema, seja a Tabela 2.6, relativa a 4 tarefas, em que se apresenta o tempo de processamento (P_i), o custo por antecipação (α_i), o custo por atraso (β_i), os tempos de preparação da máquina (S_{ij}), a data inicial (E_i) e a data final (T_i) desejadas para entrega.

Tabela 2.6: Exemplificação dos dados de entrada

	Dados do Problema					**Tempo de Preparação**			
Tarefas	P_i	E_i	T_i	α_i	β_i	1	2	3	4
1	3	3	7	9	18	0	1	1	2
2	5	11	16	10	20	1	0	3	2
3	4	5	9	7	14	1	3	0	2
4	5	7	13	8	16	2	2	2	0

Capítulo 3

Revisão Bibliográfica

Neste capítulo é feita uma revisão dos trabalhos relacionados ao problema aqui estudado.

3.1 Introdução

Nos últimos anos um grande número de empresas tem adotado a produção de bens sob encomenda através da filosofia *just-in-time*. Com a adoção deste modelo, faz-se necessário um planejamento criterioso das atividades produtivas, visto que antecipações ou atrasos podem gerar custos extras, como por exemplo, custos de armazenagem e multas contratuais, respectivamente.

Na indústria mineral esta realidade não é diferente, os prazos de entrega se tornam cada vez mais rígidos. As indústrias trabalham, normalmente, em sua capacidade máxima e com as produções já vendidas. Desta forma, são necessários métodos eficientes para planejamento do sequenciamento da produção, buscando sempre a minimização dos custos de produção, visto que antecipações ou atrasos na produção acarretam para a empresa custos, seja de estocagem ou multas contratuais, respectivamente.

O problema de sequenciamento em uma máquina com minimização das penalidades por antecipação e atraso da produção (PSUMAA), modela um dos mais simples problemas de planejamento da produção no contexto da filosofia *just-in-time*. Contudo, trata-se de um problema difícil de ser resolvido na otimalidade em tempos

computacionais aceitáveis, visto pertencer à classe NP-difícil (Du e Leung, 1990; Wan e Yen, 2002).

Vários trabalhos tratam o PSUMAA com datas de entrega. Gordon *et al.* (2002) apresenta um estudo sobre datas comuns de entrega em problemas de sequenciamento em uma máquina e em máquinas paralelas. De acordo com o autor a data de entrega se tornou um fator importante a partir de meados da década de 80 devido à filosofia *just-in-time*; porém Jackson (1955), *apud* Gordon *et al.* (2002), foi o ponto de partida sobre seu estudo. Vários métodos de formulação da função objetivo foram apresentados, além de algumas propriedades para datas comuns de entrega.

Coleman (1992) resolveu o PSUMAA com data de entrega distinta e tempo de preparação dependente da sequência através de um modelo de programação inteira mista. Em seu trabalho, Coleman (1992) resolveu problemas-teste utilizando o otimizador LINDO, com 4 e 8 tarefas.

Lee e Choi (1995) aplicaram Algoritmos Genéticos (AG) ao PSUMAA com datas de entrega distintas. Para determinar a data ótima de conclusão de processamento de cada tarefa da sequência produzida pelo AG, desenvolveram um algoritmo específico, de complexidade polinomial, que explora as características do problema.

Sridharan e Zhou (1996) estudaram o PSUMAA dinâmico com datas de entrega distintas. De acordo com os autores, para problemas com tempos de entrega distintos a solução ótima pode conter tempo ocioso entre as tarefas. A maioria dos trabalhos que estudam este problema utilizam um algoritmo de duas fases para resolver o problema. Na primeira fase é gerada uma sequência de produção e, na segunda, é calculada a data ótima de execução, podendo-se inserir tempo ocioso entre as tarefas. Esses autores, no entanto, propuseram uma heurística de uma única fase baseado em teoria de decisão para construção do sequenciamento com tempo ocioso embutido.

James (1997) resolveu o PSMUAA com data comum de entrega sem tempo ocioso entre as tarefas utilizando a metaheurística Busca Tabu. O autor utilizou os

seguintes movimentos para explorar o espaço de busca: troca entre tarefas adjacentes, troca entre duas tarefas quaisquer e inserção de uma tarefa à frente de outro. Experimentos mostraram que a troca adjacente obteve os piores resultados e um movimento híbrido de troca e inserção obteve os melhores resultados.

Li (1997), Liaw (1999) e Biskup e Feldmann (2001) trataram o PSUMAA com datas comuns de entrega, sem permitir tempo ocioso de máquina. O primeiro autor decompõe o problema em dois subproblemas com uma estrutura mais simples, de forma que o limite inferior do problema é a soma dos limites inferiores desses dois subproblemas. O limite inferior de cada subproblema é determinado por relaxação lagrangiana. Um algoritmo *branch-and-bound* é apresentado e usado para resolver instâncias de até 50 tarefas, dobrando a dimensão de problemas que podiam ser resolvidos na otimalidade com algoritmos exatos até aquela data. O autor propôs, também, procedimentos heurísticos baseados em busca local para resolver problemas de dimensões mais elevadas. No segundo trabalho é apresentado um algoritmo *branch-and-bound* que faz uso de procedimentos para determinar limites inferiores e superiores fortes. Regras de dominância são usadas para tentar eliminar nós não promissores na árvore de busca. É analisado o desempenho do algoritmo para resolver problemas de até 50 tarefas. O terceiro trabalho apresenta duas heurísticas para resolver o problema, porém o foco do trabalho está em apresentar um algoritmo para geração de problemas-teste, com objetivo principal de gerar instâncias que pudessem ser utilizadas na comparação de performance de diferentes métodos de resolução do problema.

Chang (1999) tratou o PSUMAA com datas distintas de entrega por meio de um algoritmo *branch-and-bound*. É analisado o desempenho desse algoritmo para resolver problemas com até 45 tarefas. Também é desenvolvido um esquema para delimitar o cálculo de diferentes limites inferiores baseados no procedimento de eliminação de sobreposição de uma sequência *just-in-time*. Propriedades e teoremas sobre procedimento de eliminação de sobreposição são apresentados.

Gagné *et al.* (2001) resolveram o PSUM com tempo de preparação dependente da sequência de produção com penalização por atraso utilizando a metaheu-

rística Colônia de Formigas (*Ant Colony Optimization* - ACO). O método ACO proposto foi adaptado do método ACO utilizado para resolver o PCV.

Feldmann e Biskup (2003) resolveram o PSUMAA com datas comuns de entrega por meio de três métodos: um algoritmo evolutivo, *Simulated Annealing* (SA) e uma versão aperfeiçoada da heurística *Threshold Accepting*, sendo esta última uma variante de *Simulated Annealing*. O mesmo problema foi tratado por Hino *et al.* (2005), os quais apresentaram métodos baseados nas metaheurísticas Busca Tabu e Algoritmos Genéticos, bem como hibridizações destas.

Ventura e Radhakrishnan (2003) estudaram o PSUMAA com o tempo de preparação incluído no tempo de processamento. Os autores apresentaram algumas propriedades e desenvolveram uma heurística para encontrar uma solução inicial e então resolveram o problema por meio de relaxação lagrangeana com um algoritmo de subgradiente.

Rabadi *et al.* (2004) focaram no PSUMAA com datas comuns de entrega e tempo de preparação da máquina dependente da sequência de produção. Os autores apresentam um procedimento *branch-and-bound* que é capaz de resolver na otimalidade, em tempos aceitáveis, problemas-teste de até 25 tarefas, o que representava um avanço porque até aquela data algoritmos exatos para essa classe de problemas eram capazes de resolver somente problemas-teste de até 8 tarefas.

Tian *et al.* (2005) mostraram que, fazendo uso de algumas propriedades, o problema de sequenciamento em uma máquina com minimização do tempo total de atraso e datas de entrega distintas pode ser resolvido em tempo polinomial usando o algoritmo *shortest processing time* (SPT), para um determinado número de datas de entrega distintas.

Hassin e Shani (2005) apresentaram um estudo sobre o PSUMAA com datas de entrega distintas, com algumas variações, tempos de processamentos comuns e distintos, além de um novo conceito denominado não-execução, onde uma tarefa pode não ser executada, atribuindo portanto uma penalidade por não-execução. Para resolver o problema os autores adaptaram um algoritmo de tempo polinomial

da literatura, por eles denominado GTW.

Gupta e Smith (2006) propuseram dois métodos, um baseado em GRASP e outro na heurística *space-based search*, para a resolução do problema de sequenciamento em uma máquina considerando a existência de data de entrega para cada tarefa e tempo de preparação de máquina dependente da sequência de produção, tendo como objetivo a minimização do tempo total de atraso. O método GRASP proposto foi dividido em três fases: Construção, Refinamento e Reconexão por Caminhos. Segundo os autores, a contribuição está em uma nova função custo para a fase de construção, uma nova variação do método VND para a fase de refinamento e uma fase de Reconexão por Caminhos usando diferentes vizinhanças.

Os autores Lin *et al.* (2007a,b) também estudaram o PSUMAA com datas comuns de entrega. No primeiro trabalho foram utilizadas as metaheurísticas Algoritmos Genéticos (AG) e *Simulated Annealing* (SA), nas quais é utilizada uma busca local gulosa. No segundo, os autores desenvolveram uma heurística baseada em troca sequencial. Em ambos, os autores fizeram uso de três propriedades da literatura aplicadas ao problema com datas comuns de entrega para melhorar o desempenho das heurísticas.

Hallah (2007) trata do PSUMAA com datas de entrega distintas, não sendo permitida a existência de tempo ocioso entre as tarefas. O autor propõe um método híbrido que combina heurísticas de busca local (regras de despacho, método da descida e *Simulated Annealing*) e um algoritmo evolucionário.

O PSUM com penalidade por atraso e data de entrega distinta foi estudado por Bilge *et al.* (2007), os quais utilizaram a metaheurística Busca Tabu para resolvê-lo. Para gerar a solução inicial os autores utilizaram vários métodos, entre eles as heurísticas *earliest due date* (EDD) e *weighted shortest processing time* (WSPT). A BT desenvolvida usou uma estrutura de vizinhança híbrida com movimentos de troca e realocação, porém os autores utilizaram uma estratégia para criação de uma lista de candidatos, se valendo de algumas propriedades do problema, para não permitir movimentos que não levariam a boas soluções. Foi também utilizado um tempo tabu (*tabu tenure*) dinâmico para evitar a ciclagem.

Ying (2008) trata do PSUMAA com datas comuns de entrega das tarefas e com tempo de *setup* de cada tarefa incluído em seu tempo de processamento e independente da sequência de produção. O problema é resolvido pelo algoritmo *Recovering Beam Search* (RBS), que é uma versão aperfeiçoada do algoritmo *Beam Search* (BS). Este, por sua vez, consiste em um algoritmo *branch-and-bound* em que somente os w nós mais promissores de cada nível da árvore de busca são retidos para ramificação futura, enquanto os nós restantes são podados permanentemente. Para evitar decisões equivocadas com respeito à poda de nós que conduzam à solução ótima, o algoritmo RBS utiliza uma fase de recobrimento que busca por soluções parciais melhores que dominem aquelas anteriormente selecionadas.

Alguns trabalhos, como Chen (1997) e Uzsoy e Velásquez (2008), tratam o tempo de preparação dependente da sequência por famílias, onde cada família consiste em um conjunto de tarefas que não possuem tempo de preparação entre si. O tempo de preparação somente existe entre as execuções das famílias de tarefas. Uzsoy e Velásquez (2008) utilizaram várias heurísticas para resolver o problema, entre elas uma baseada unicamente na heurística EDD e outra, baseada em EDD com uma busca local.

Wan e Yen (2002) apresentaram um método baseado na metaheurística Busca Tabu (BT) para resolver o PSUMAA com janelas de entrega distintas. Para cada sequência de tarefas gerada pela Busca Tabu é acionado um procedimento de complexidade polinomial para determinar a data de conclusão ótima do processamento de cada tarefa da sequência. Este último procedimento é adaptado daquele proposto em Lee e Choi (1995), em que se consideram janelas de entrega em lugar de datas de entrega.

Gomes Jr. *et al.* (2007) desenvolveram um modelo de programação linear inteira mista para o PSUMAA com janelas de entrega e tempo de preparação dependente da sequência de produção. Esta aplicação foi inspirada na dissertação de Bustamante (2006). Neste último trabalho, considera-se uma aplicação na indústria siderúrgica, em que uma máquina é considerada como sendo uma sequência de laminadores que representam o gargalo do sistema, e cada tarefa representa um conjunto

de operações realizadas nesses laminadores que resulte em um dado produto (cantoneira, perfil U, vergalhão etc.). Para cada produto fabricado têm-se um conjunto de operações de mesma natureza realizadas nos mesmos laminadores, mas que requerem tempos de processamento diferentes, bem como ajustes na configuração dos equipamentos da laminação. Como cada ajuste nos equipamentos dura um tempo para ser executado, a composição destes tempos gera o tempo para troca entre cada produto.

O modelo de Gomes Jr. *et al.* (2007) foi utilizado para resolver na otimalidade problemas de até 12 tarefas. Esta modelagem serviu para comparar os resultados obtidos por um algoritmo heurístico baseado em GRASP, ILS e VND, também proposto pelos autores. Para cada sequência gerada pela heurística, é acionado um algoritmo para determinar a data ótima de conclusão do processamento de cada tarefa. Tal algoritmo foi adaptado de Wan e Yen (2002), e inclui no tempo de processamento de uma tarefa o tempo de preparação da máquina, já que quando ele é acionado, já se conhece a sequência de produção. O algoritmo desenvolvido pelos autores foi capaz de encontrar todas as soluções ótimas conhecidas.

Uma revisão mais abrangente de trabalhos sobre sequenciamento com tempos de preparação dependentes da sequência pode ser encontrada em Allahverdi *et al.* (1999) e Allahverdi *et al.* (2008).

Capítulo 4

Metodologia

Neste capítulo são apresentadas duas metodologias de solução do problema tratado. Na seção 4.1, é apresentado um modelo de programação matemática para resolver na otimalidade problemas-teste do PSUMAA de pequenas dimensões. Na seção 4.3, é apresentado um algoritmo heurístico para encontrar soluções sub-ótimas do PSUMAA.

A utilização de procedimentos heurísticos para resolver o PSUMAA se deve à complexidade combinatória do problema tratado. O algoritmo proposto baseia-se em metaheurísticas, as quais, ao contrário dos procedimentos heurísticos clássicos, têm mecanismos que possibilitam escapar das armadilhas dos ótimos locais. Mais especificamente, propõe-se um algoritmo de três fases, denominado GTSPR, combinando os procedimentos metaheurísticos GRASP (Feo e Resende, 1995), VND (Mladenović e Hansen, 1997), Busca Tabu (Glover e Laguna, 1997) e a técnica Reconexão por Caminhos (Glover, 1996).

Na primeira fase, é construída uma solução inicial utilizando-se GRASP e VND. Na segunda fase essa solução é refinada por um procedimento baseado em Busca Tabu. Na terceira e última fase, como pós-otimização, é aplicado Reconexão por Caminhos a pares de soluções elite produzidas durante a Busca Tabu.

A fase GRASP-VND do algoritmo envolve dois estágios, construção e refinamento, os quais são aplicados consecutivamente por várias iterações. No estágio de construção, uma solução é construída passo a passo, elemento por elemento. Para

decidir qual elemento deve ser inserido na solução em construção é feita uma lista restrita de candidatos (LRC) com os melhores elementos candidatos. Esses candidatos são avaliados segundo uma certa função que leva em consideração os elementos já inseridos e os benefícios com a inserção do elemento candidato. A seguir, um dos elementos da LRC é escolhido de acordo com uma certa probabilidade, normalmente igual para todos da LRC. Terminado o estágio de construção, a solução construída é refinada pelo procedimento VND.

A vantagem do procedimento GRASP é que ele combina uma característica positiva dos métodos construtivos puramente gulosos (a boa qualidade da solução construída, facilitando o refinamento), com uma característica positiva dos procedimentos totalmente aleatórios (a diversificação do espaço de busca). Como construir uma solução para o PSUMAA é um procedimento relativamente rápido, demandando pouco esforço computacional, e refinar a solução construída tem um alto custo computacional, propõe-se fazer o refinamento mais "leve" após cada construção, seguido de um refinamento mais "pesado" na melhor das soluções construídas e refinadas, na fase GRASP-VND do algoritmo.

Na segunda fase, a solução proveniente da fase GRASP-VND é refinada pelo procedimento metaheurístico Busca Tabu (Glover, 1986). A Busca Tabu é um método de otimização, baseado em busca local, que admite soluções de piora para escapar das armadilhas dos ótimos locais e utiliza uma estrutura de memória para não retornar à soluções já geradas. Em sua forma original, a cada iteração, da Busca Tabu, procura-se um ótimo local selecionando-se o melhor vizinho s' da vizinhança $N(s)$ da solução corrente s. Independentemente de $f(s')$ ser melhor ou pior que $f(s)$, s' será sempre a nova solução corrente. Entretanto, este mecanismo, por si só, não é suficiente para escapar de ótimos locais, uma vez que pode haver retorno a uma solução previamente gerada. Para evitar isso, o algoritmo usa o conceito de lista tabu. Esta lista define todos os movimentos executados, baseados em um certo atributo, como sendo tabu por um determinado número de iterações, conhecido como tempo tabu (*tabu tenure*). Tais movimentos são proibidos a menos que a solução satisfaça a um certo critério de aspiração, em geral que essa solução seja melhor que

a melhor solução encontrada até então. Os atributos são escolhidos para prevenir o retorno a soluções visitadas recentemente e são escolhidos por características fáceis de detectar.

Por fim, na terceira e última fase do algoritmo proposto, é utilizado o procedimento de Reconexão por Caminhos (RC) como estratégia de pós-refinamento. Proposto por Glover (1996), a RC é uma estratégia de intensificação normalmente utilizada como pós-otimização de uma solução, ou refinamento de ótimos locais obtidos durante a busca. Dado um par de soluções, o objetivo da técnica é partir de uma delas, dita solução base, e chegar à outra, dita solução guia, por meio da adição na solução base de atributos da solução guia.

O algoritmo heurístico proposto, após gerar uma sequência de produção, utiliza um procedimento para programá-la, isto é, determinar as datas ótimas de inicio e término de cada tarefa na sequência gerada. Esse procedimento está descrito na seção 4.2. A Figura 4.1 descreve o modelo de obtenção da solução para o PSUMAA abordado.

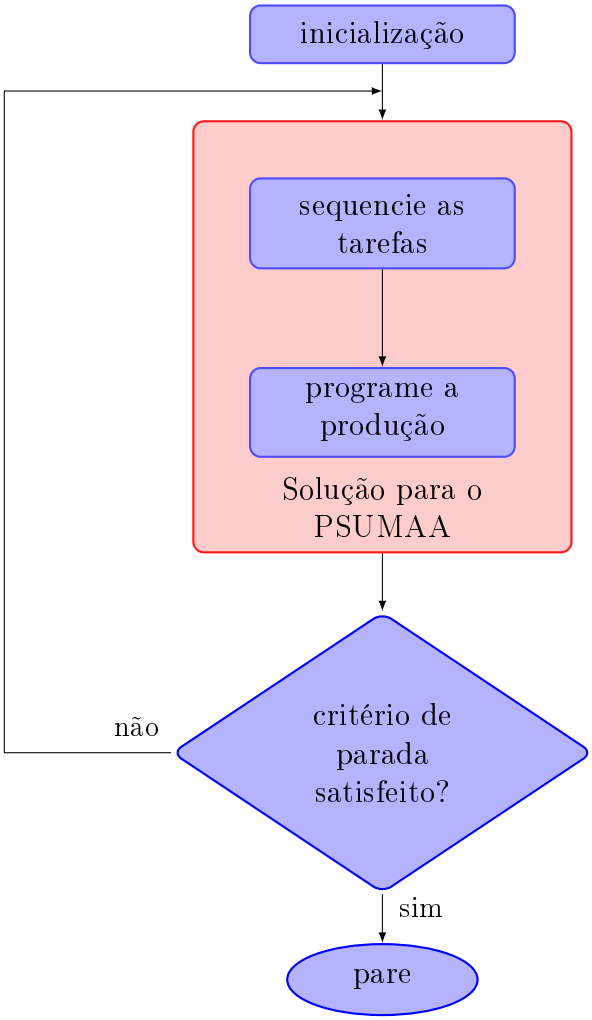

Figura 4.1: Metodologia heurística de solução para o PSUMAA

4.1 Modelo Matemático

O modelo matemático apresentado a seguir, relativo ao PSUMAA abordado, foi desenvolvido por Gomes Jr. *et al.* (2007), e é aqui reproduzido.

Nesse modelo são utilizadas duas tarefas fictícias, 0 (zero) e $n+1$, de tal forma que a tarefa 0 antecede imediatamente a primeira operação e a tarefa $n+1$ sucede imediatamente a última na sequência de produção. Considera-se M uma constante de valor suficientemente grande, P_0 e P_{n+1} iguais a zero e $S_{0i} = 0$ e $S_{i,n+1} = 0$, $\forall i = 1, ..., n$.

As variáveis de decisão do modelo são:

- s_i, representando a data de início do processamento da tarefa $i (s_i \geq 0)$;
- e_i, indicando o tempo de antecipação da tarefa i;
- t_i, indicando o tempo de atraso da tarefa i e
- y_{ij}, uma variável binária definida da seguinte forma:

$$y_{ij} = \begin{cases} 1, & \text{se a tarefa } j \text{ é sequenciada imediatamente após a tarefa } i; \\ 0, & \text{caso contrário.} \end{cases}$$

O modelo de programação linear inteira mista de Gomes Jr. *et al.* (2007) é definido pelas expressões (4.1) a (4.10), a seguir:

minimizar:

$$\sum_{i=1}^{n} (\alpha_i e_i + \beta_i t_i) \tag{4.1}$$

sujeito a:

$$s_j - s_i - y_{ij}(M + S_{ij}) \geq P_i - M \qquad \forall i = 0, 1, \cdots, n \quad \forall j = 1, 2, \cdots, n+1 \text{ e } i \neq j \tag{4.2}$$

$$\sum_{i=0, i \neq j}^{n} y_{ij} = 1 \qquad \forall j = 1, 2, \cdots, n+1 \tag{4.3}$$

$$\sum_{j=1, j \neq i}^{n+1} y_{ij} = 1 \qquad \forall i = 0, 1, \cdots, n \tag{4.4}$$

$$s_i + P_i + e_i \geq E_i \qquad \forall i = 1, 2, \cdots, n \tag{4.5}$$

$$s_i + P_i - t_i \leq T_i \qquad \forall i = 1, 2, \cdots, n \tag{4.6}$$

$$s_i \geq 0 \qquad \forall i = 0, 1, \cdots, n+1 \tag{4.7}$$

$$e_i \geq 0 \qquad \forall i = 1, 2, \cdots, n \tag{4.8}$$

$$t_i \geq 0 \qquad \forall i = 1, 2, \cdots, n \tag{4.9}$$

$$y_{ij} \in \{0, 1\} \qquad \forall i, j = 0, 1, \cdots, n+1 \tag{4.10}$$

A função objetivo, representada pela equação (4.1), tem como critério de otimização a minimização do custo total de antecipação e atraso. As restrições (4.2) definem a sequência de operações na máquina utilizada, garantindo a existência de um tempo suficiente para completar uma tarefa i antes de se começar outra tarefa j. Quando $y_{ij} = 1$, as restrições (4.2) se reduzem a $s_j \geq s_i + S_{ij} + P_i$, isto é, a tarefa j só pode começar após iniciada a tarefa predecessora i, processado-a e preparada a máquina para executar a tarefa j. Quando $y_{ij} = 0$, as restrições (4.2) se reduzem a $s_j \geq s_i + P_i - M$, e portanto, são redundantes, pois M é uma constante de valor arbitrariamente grande. As restrições (4.4) e (4.3) garantem que cada tarefa tenha somente uma tarefa imediatamente antecessora e uma tarefa imediatamente sucessora, respectivamente. As restrições (4.5) e (4.6) definem a quantidade de atraso e de antecipação de cada tarefa i. As restrições (4.7) a (4.10) definem o tipo das variáveis do problema.

4.2 Procedimento de Determinação das Datas Ótimas de Início de Processamento das Tarefas

Dada uma sequência de tarefas, o procedimento de determinação das datas ótimas de início de processamento (PDDOIP) determina a melhor data para se iniciar o processamento de cada tarefa, de acordo com a janela de entrega da mesma.

O PDDOIP implementado é o utilizado em Gomes Jr. *et al.* (2007), o qual foi baseado em Wan e Yen (2002) e Lee e Choi (1995), incorporando-se o tempo de preparação S_{ij} ao tempo de processamento da tarefa j. Tal consideração pode ser feita uma vez que, para sua aplicação, a sequência de produção é conhecida.

Sejam V a sequência de tarefas dada e C_k a data de conclusão do processamento da tarefa k. Suponha que um subconjunto de tarefas $B_j = \{J_0, J_1, ..., J_m\} \subseteq V$ forme um bloco no sequenciamento, ou seja, as tarefas em B_j são sequenciadas consecutivamente sem tempo ocioso entre elas. Além disso, assuma que há l blocos em V e sejam $prim(j)$ e $ult(j)$ a primeira e a última tarefa no bloco B_j, respectivamente.

O processamento da primeira tarefa (J_1) da sequência V é programado para ser finalizado na data E_{J_1} se $P_{J_1} \leq E_{J_1}$ ou iniciado na data 0 (finalizando na data P_{J_1}) se $P_{J_1} > E_{J_1}$.

As demais tarefas são programadas da seguinte forma:

- Se $C_{J_k} + S_{(J_k)(J_{k+1})} + P_{J_{k+1}} < E_{J_{k+1}}$, então a tarefa J_{k+1} tem seu processamento programado para ser finalizado na data E_{Jk+1}, isto é, a tarefa J_{k+1} tem seu processamento programado para ser finalizado no início de sua janela de entrega, desta forma, iniciando um novo bloco.
- Por outro lado, se $C_{J_k} + S_{(J_k)(J_{k+1})} + P_{J_{k+1}} \geq E_{J_{k+1}}$, então a tarefa J_{k+1} tem seu processamento programado para ser finalizado na data $C_{J_k} + S_{(J_k)(J_{k+1})} + P_{J_{k+1}}$ e é incluído como último elemento do bloco corrente.

Após a determinação da data de conclusão do processamento de cada tarefa, é necessário verificar o posicionamento dos blocos. O custo mínimo do bloco B_j ocorre nos pontos extremos de sua função custo, isto é, no início ou no final da janela de entrega de uma das tarefas no bloco. Por causa da natureza linear convexa por partes da função custo, o custo mínimo pode ser facilmente obtido pela comparação dos somatórios das penalidades das tarefas no bloco, no início e no final de cada janela de entrega no bloco (penalidades por antecipação são consideradas negativas enquanto penalidades por atraso são consideradas positivas). Estes somatórios fornecem as inclinações das retas (m) pertencentes à função custo. O custo mínimo do bloco B_j se dá no ponto extremo onde a inclinação m torna-se maior ou igual a zero. Quando o ponto mínimo do bloco B_j é encontrado, todo o bloco é movido em direção ao ponto mínimo até que um dos três casos seguintes aconteça:

1. $s_{prim(j)} = 0$.
2. O ponto mínimo é alcançado.
3. $s_{prim(j)}$ torna-se igual a $C_{ult(j-1)} + S_{(ult(j-1))(prim(j))}$.

No caso (3), o bloco B_j é unido ao bloco B_{j-1}, resultando em um novo bloco B_{j-1}. Então, o ponto mínimo do novo B_{j-1} deve ser obtido. O procedimento anterior deve ser realizado até que o caso (1) e (2) ocorram para cada bloco.

Proposição 1 (Wan e Yen, 2002) - *O custo total de uma dada sequência de tarefas alcança seu valor ótimo se cada bloco B_j na sequência alcançar seu ponto mínimo, com exceção de $s_{prim(1)}$, que pode ser igual a zero para o primeiro bloco.*

Para exemplificar o método, serão utilizados os dados da Tabela 2.6, apresentada na página 18 da seção 2.3, relativa a 4 tarefas.

Dada a sequência $V = \{3, 4, 1, 2\}$, a primeira tarefa da sequência, tarefa 3, denotada por J_1, tem seu processamento programado para ser finalizado na data $E_3 = 5$, que representa o início da sua janela de entrega. Para calcular as inclinações m da reta que indica o ponto de mínimo do bloco para o primeiro elemento, a penalidade por antecipação ($\alpha_3 = 7$) assume valor negativo e a por atraso ($\beta_3 = 14$) assume valor positivo. A Figura 4.2 descreve o primeiro passo da programação de tarefas.

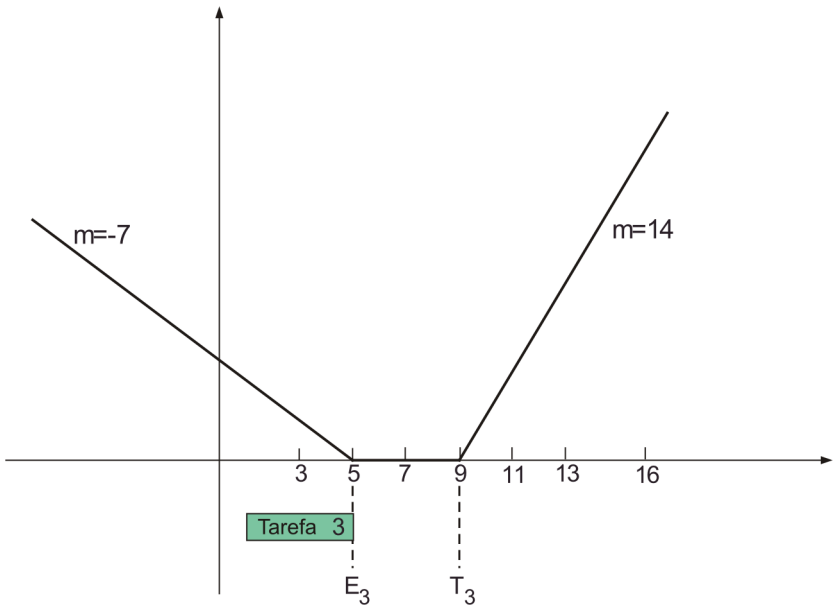

Figura 4.2: Programação das tarefas de um bloco na sequência: 1^o Passo

A Figura 4.3 exemplifica a programação da segunda tarefa da sequência, tarefa $J_2 = 4$, que tem sua conclusão programada para ser finalizada no instante

$C_4 = 12$, pois $C_3 + S_{34} + P_4 > E_4$. A inclinação m da reta, para instantes inferiores a -4, é dado por $m = -7 - 8 = -15$, momento que todas as tarefas se encontram adiantadas. Entre os instantes -4 e 1, somente a tarefa 3 está adiantada, portanto $m = -7$. A partir do instante 2 as tarefas passam a ficar atrasadas, acarretando em uma inclinação positiva. Até o instante 3 somente a tarefa 4 está atrasada, sendo $m = 16$. Após este instante as tarefas 3 e 4 tornam-se atrasadas, portanto $m = 14 + 16 = 30$ a partir do instante 3.

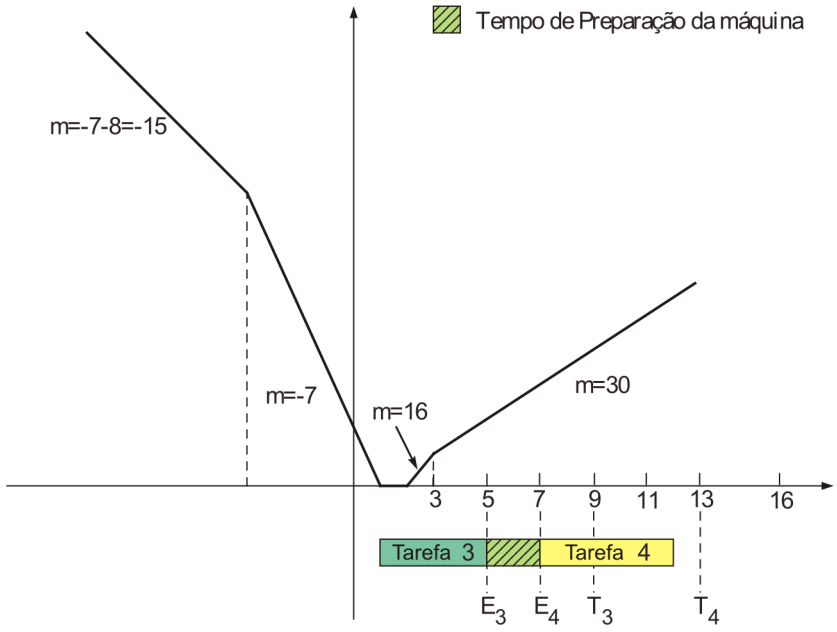

Figura 4.3: Programação das tarefas de um bloco na sequência: 2^o Passo

Na Figura 4.4 são apresentadas as tarefas 3 e 4 já programadas, e mostra-se como determinar a data de conclusão do processamento da tarefa 1, os valores das inclinações m e o ponto mínimo do bloco. Até este momento, a tarefa $J_1 = 3$ se inicia no instante 1 e a tarefa $J_2 = 4$ no instante 7, ambos terminando dentro de sua janela de tempo. Como $C_{J_2} + S_{(J_2)(J_3)} + P_{J_3} = 17 \geq E_{J_3} = 3$, então $C_{J_3} = C_1 = 17$, isto é, a tarefa $J_3 = 1$ deve ser concluída no instante 17 e iniciada no instante dado por $s_{J_3} = s_1 = 17 - 3 = 14$. Neste caso, a tarefa 1 é inserida como último elemento do bloco corrente, conforme pode ser observado nesta figura. Conhecida a data de conclusão da tarefa 1, deseja-se agora verificar a posição do bloco, para saber se ele está em seu ponto mínimo. Para isto, primeiramente, são determinadas as possíveis datas de início do processamento do bloco. Essas datas são obtidas posicionando

cada tarefa do bloco no início e no final de sua janela de entrega, pois como foi visto, o custo mínimo do bloco se dá no início ou no final da janela de entrega de uma das tarefas no bloco.

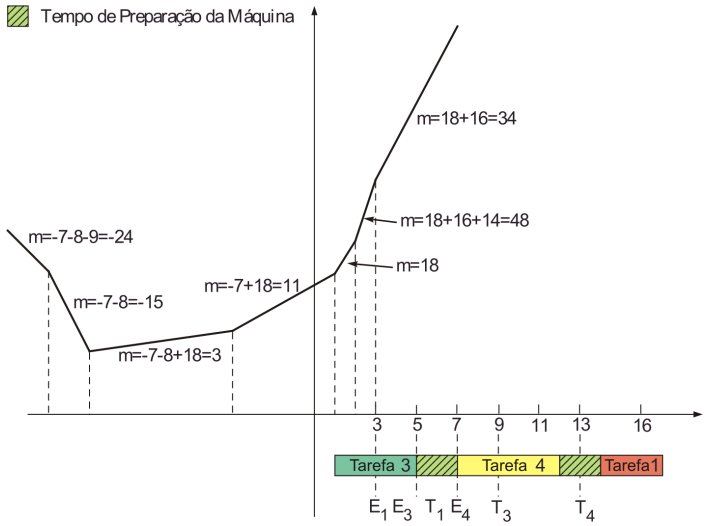

Figura 4.4: Programação das tarefas de um bloco na sequência: 3^o Passo

No exemplo mostrado, as possíveis datas de início do processamento do bloco são: $-13, -11, -4, 1, 2$ e 5. Determinadas as possíveis datas de início de processamento, calculam-se então as inclinações m de acordo com essas datas. No exemplo apresentado, se o processamento do bloco iniciar antes da data -13, ou seja, com todas as tarefas sendo antecipadas, então a inclinação m é dada pelo somatório das penalidades por antecipação das tarefas 3, 4 e 1, portanto $m = -7-8-9 = -24$. Agora, se o processamento do bloco iniciar entre a data - 13 e - 11, a tarefa 1 deixa de estar antecipada, e a inclinação m passa a ser $m = -7-8 = -15$ (somatório das penalidades por antecipação das tarefas 3 e 4). Entre a data - 11 e - 4 a tarefa 1 passa a estar atrasada, então a inclinação m é dada pelo somatório das penalidades por antecipação das tarefas 3 e 4 e da penalidade por atraso da tarefa 1, ou seja, $m = -7-8+18 = 3$. Este processo se repete até a última data. Graficamente, este procedimento está representado na Figura 4.4. O ponto mínimo, o qual se deseja encontrar, se dá no ponto aonde a inclinação m se torna maior ou igual a zero. No exemplo, esse ponto corresponde à data -11. Determinado o ponto de mínimo, então todo o bloco deve ser movido para a esquerda até que um dos três casos citados anteriormente ocorra. No exemplo considerado, o bloco é empurrado

até o caso (1) ocorrer, ou seja, $s_{prim(1)} = 0$ (Figura 4.5).

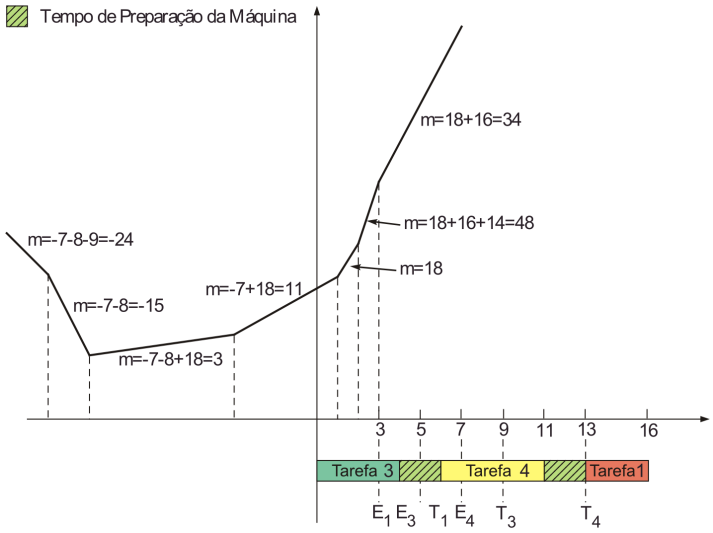

Figura 4.5: Programação das tarefas de um bloco na sequência: 4^o Passo

O PDDOIP para uma sequência de tarefas dadas é descrito na Figura 4.6, onde o procedimento MudaBloco (Linha 26 da Figura 4.6) movimenta o bloco B_j até seu ponto de mínimo e, caso seja necessário, combina-o com o bloco anterior.

Procedimento `DetermineDatasOtimasInicio(`n, v, $f(v)$`)`

```
1  início
2      B ← 1                                                    // Inicialização
3      prim(B) ← 1
4      ult(B) ← 1
5      s_1 ← max{0, E_1 − P_1}
6      C_1 ← max{E_1, P_1}                                      // Demais Tarefas
7      para i = 2 até n faça
8          se (C_{i−1} + P_i + S_{(i−1)(i)} < E_i) então
9              B ← B + 1
10             prim(B) ← 1
11             ult(B) ← i
12             s_i ← E_i − P_i + S_{(i−1)(i)}
13             C_i ← E_i
14         senão
15             se (C_{i−1} + P_i + S_{(i−1)(i)} = E_i) então
16                 ult(B) ← i
17                 s_i ← C_{i−1} + S_{(i−1)(i)}
18                 C_i ← E_i
19             senão
20                 ult(B) ← i
21                 s_i ← C_{i−1} + S_{(i−1)(i)}
22                 C_i ← s_i + P_i
23             fim
24         fim
25         repita
26             MudaBloco(B)
27         até ((até todos os blocos estiverem no ponto mínimo) ou (s_{prim(1)} = 0))
28     fim
29     retorna f(v)
30 fim
```

Figura 4.6: Procedimento PDDOIP

4.3 Modelo Heurístico

4.3.1 Representação de uma Solução

Uma solução do PSUMAA é representada por um vetor v de n elementos, onde cada posição $i = 1, 2, ..., n$ indica a ordem de produção da i-ésima tarefa da sequência. Assim, na sequência $v = \{5, 3, 2, 1, 4, 6\}$, a tarefa 5 é a primeira a ser realizada e a tarefa 6, a última.

4.3.2 Vizinhança de uma Solução

Para explorar o espaço de soluções são utilizados três tipos de movimentos: troca da ordem de processamento de duas tarefas da sequência de produção, realocação de uma tarefa para outra posição na sequência de produção e realocação de um bloco de k tarefas, $2 \leq k \leq n - 2$. Esses movimentos definem, respectivamente, as estruturas de vizinhança N^T, N^R e N^{Or}.

Dado um conjunto com n tarefas, há $n(n-1)/2$ vizinhos possíveis na vizinhança N^T. Por exemplo, a solução $v' = \{5, 4, 2, 1, 3, 6\}$ é um vizinho da solução v na vizinhança N^T, pois é obtido pela troca da tarefa 3, que está na segunda posição de v, com a tarefa 4, que está na quinta posição de v, como pode ser visto na Figura 4.7.

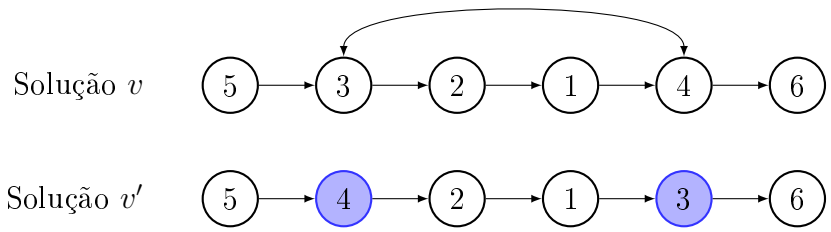

Figura 4.7: Movimento de troca - N^T

Na segunda estrutura de vizinhança, N^R, há $(n-1)^2$ vizinhos possíveis. Por exemplo, a solução $v' = \{5, 2, 1, 4, 3, 6\}$ é vizinha de v na vizinhança N^R, pois é obtida pela realocação da tarefa 3 para depois da tarefa 4 na sequência de produção v, conforme exemplificado na Figura 4.8.

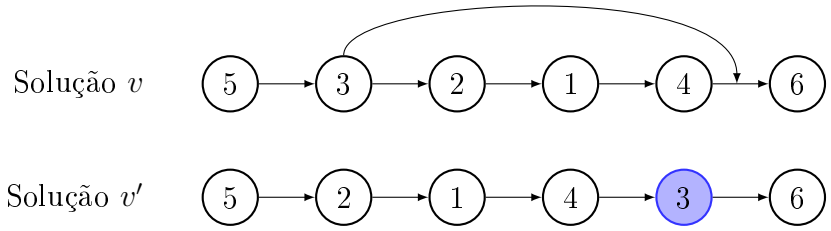

Figura 4.8: Movimento de realocação - N^R

Já na terceira estrutura de vizinhança, N^{Or}, há $(n-k-1)^2$ vizinhos possíveis. Por exemplo, a solução $v' = \{5, 2, 1, 4, 3, 6\}$ é vizinha de v na vizinhança N^{Or}, pois é obtida pela realocação do bloco de tarefas 5 e 3, com $k = 2$, para depois da tarefa 1 na sequência de produção v, mostrado na Figura 4.9. Nas vizinhanças N^R e N^{Or} são permitidos movimentos para posições sucessoras ou antecessoras àquela em que a tarefa se encontra na sequência de produção.

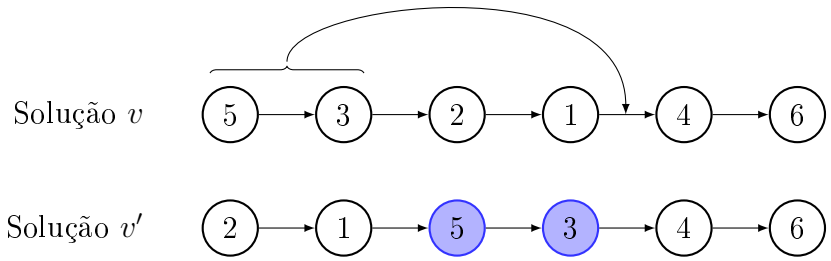

Figura 4.9: Movimento de realocação de um bloco - N^{Or}

4.3.3 Função de Avaliação

Uma solução v é avaliada pela função f a seguir, a qual deve ser minimizada:

$$f(v) = \sum_{i=1}^{n} (\alpha_i e_i + \beta_i t_i) \tag{4.11}$$

em que e_i e t_i $(e_i, t_i \geq 0)$ indicam, respectivamente, o tempo de antecipação e atraso da tarefa i e α_i e β_i são as penalidades respectivas.

Para determinar as datas ótimas de início de processamento das tarefas da

sequência dada e, consequentemente, determinar os valores e_i e t_i acima, utiliza-se o procedimento PDDOIP de Gomes Jr. *et al.* (2007).

Como os movimentos adotados não geram soluções inviáveis, pode-se observar que a função de avaliação f é a própria função objetivo do PLIM.

4.3.4 Geração da solução inicial

A solução inicial é gerada por um procedimento que combina GRASP (Feo e Resende, 1995) com Descida em Vizinhança Variável (*Variable Neighborhood Descent* - VND) (Mladenović e Hansen, 1997). A Figura 4.10 esquematiza esse procedimento.

Procedimento `GRASP-VND(`*v, γ, GRASPmax, MRDmax*`)`

```
1  início
2  |  f* ← ∞
3  |  para i=1 até GRASPmax faça
4  |  |  Escolha γ ∈ Γ
5  |  |  v ← Constroi_Solucao_GRASP(γ)
6  |  |  v' ← VND_1(v, MRDmax)
7  |  |  se (f(v') < f*) então
8  |  |  |  f* ← f(v')
9  |  |  |  v* ← v'
10 |  |  fim
11 |  fim
12 |  v ← VND_2(v*, MRDmax)
13 |  retorna v
14 fim
```

Figura 4.10: Procedimento GRASP-VND aplicado ao PSUMAA

Na fase de construção (Linha 5 da Figura 4.10), uma solução é formada gradativamente, sendo que a cada passo é adicionada uma única tarefa à sequência. Para se escolher a tarefa a ser adicionada é feita uma lista de candidatos (LC) com todas as tarefas ainda não sequenciadas. Essas são ordenadas pela data de início da janela de entrega de cada tarefa, sendo a com a data mais cedo (E_{min}) a primeira da lista e a de data mais tarde (E_{max}) a última, tal como na heurística EDD (*Earliest Due Date*) (Baker e Scudder, 1990).

A partir de LC é formada uma lista restrita de candidatos (LRC), com

as tarefas melhor classificadas segundo o critério da data de início da janela de entrega das tarefas. O tamanho dessa lista restrita de candidatos é definido por um parâmetro $\gamma \in [0, 1]$, escolhido em um conjunto Γ, tal como em Gomes Jr. *et al.* (2007). Fazem parte da LRC todas as tarefas i cujas datas E_i sejam menores ou iguais à $E_{min}+\gamma(E_{max}-E_{min})$. A seguir, é escolhida aleatoriamente uma tarefa dessa lista, sendo a mesma adicionada à solução parcial. A fase de construção é encerrada quando todas as tarefas forem alocadas. Este procedimento é apresentado na Figura 4.11.

Procedimento `Constroi_Solucao_GRASP(`v, γ`)`

```
1  início
2  |  v ← ∅
3  |  Inicialize o conjunto LC de tarefas candidatas
4  |  enquanto (LC ≠ ∅) faça
5  |  |  E_min = min_{i ∈ LC} {E_i}
6  |  |  E_max = max_{i ∈ LC} {E_i}
7  |  |  LRC = {i ∈ LC | E_i ≤ E_min + γ(E_max − E_min)}
8  |  |  Selecione, aleatoriamente, uma tarefa i ∈ LRC
9  |  |  v ← v ∪ {i}
10 |  |  Atualize o conjunto LC de tarefas candidatas
11 |  fim
12 |  retorna v
13 fim
```

Figura 4.11: Procedimento Constroi_Solucao_GRASP aplicado ao PSUMAA

Na fase de refinamento do procedimento GRASP-VND (Linha 6 da Figura 4.10) é aplicado o procedimento VND_1 (Figura 4.12) a cada solução construída. Essa heurística de refinamento explora o espaço de soluções usando as vizinhanças N^R e N^T, nesta ordem. Inicialmente é feita uma descida randômica usando-se movimentos de realocação. Para tanto, é escolhida aleatoriamente uma tarefa e uma posição para inserí-la. Se esta nova sequência produzir uma solução com um valor menor para a função de avaliação, a nova sequência é aceita e passa a ser a solução corrente; caso contrário, é testada outra realocação aleatória. A fase de realocação termina quando houver *MRDmax* iterações consecutivas sem melhora na função de avaliação. Nesta última situação, passa-se a fazer movimentos de troca, também de forma aleatória.

Se for produzida uma solução global de melhor qualidade, interrompe-se esta fase e volta-se para a fase de realocação; caso contrário, prosseguem-se com as trocas. O procedimento é encerrado quando não houver melhora na solução global nem com movimentos de realocação nem com movimentos de troca.

Procedimento $\texttt{VND}_1(v,\ MRDmax)$

```
1  início
2      enquanto (melhorou) faça
3          melhorou ← falso
4          iter ← 0
5          enquanto (iter ≤ MRDmax) faça
6              Encontre um vizinho aleatório v' ∈ N^R(v)
7              se (f(v') < f(v)) então
8                  v ← v'
9                  melhorou ← verdadeiro
10                 iter ← 0
11             fim
12             iter = iter + 1
13         fim
14         se (melhorou) então
15             iter ← 0
16             enquanto (iter ≤ MRDmax) faça
17                 Encontre um vizinho aleatório v' ∈ N^T(v)
18                 se (f(v') < f(v)) então
19                     v ← v'
20                     Interrompa esta fase e volte para a fase de realocação
21                 fim
22                 iter = iter + 1
23             fim
24         fim
25     fim
26     retorna v
27 fim
```

Figura 4.12: Procedimento VND_1 aplicado ao PSUMAA

Deve-se observar que a solução resultante desta busca não é necessariamente um ótimo local com relação às vizinhanças consideradas, visto que a vizinhança completa não é analisada a cada iteração. Em virtude desse fato, decorridas *GRASPmax* iterações, a melhor das soluções obtidas é submetida a novo refinamento (Linha 12 da Figura 4.10), desta vez pelo procedimento VND_2. Este procedimento consiste

em explorar o espaço de soluções usando-se, além dos movimentos tradicionais de realocação e troca, também o movimento Or descrito na Subseção 4.3.2.

No procedimento VND_2, a busca é feita de acordo com a seguinte estratégia de exploração do espaço de busca:

1. descida randômica com movimentos de realocação;
2. descida randômica com movimentos de troca;
3. descida randômica com movimentos Or de realocação de um bloco de k tarefas $(2 \leq k \leq 3)$;
4. descida completa com movimentos de realocação;
5. descida completa com movimentos de troca;
6. descida completa com movimentos Or de realocação de um bloco de k tarefas $(2 \leq k \leq n-2)$.

Nas descidas com realocação de um bloco de k tarefas, k assume inicialmente seu valor mais baixo. Havendo alguma melhora, passa-se para a primeira estratégia; caso contrário, k é aumentado progressivamente até atingir seu valor máximo. Em todas as descidas randômicas para-se quando houver $MRDmax$ iterações sem melhora. Observa-se que o procedimento VND_2 retorna um ótimo local com respeito às vizinhanças N^R, N^T e N^{Or}.

4.3.5 Busca Tabu aplicada ao PSUMAA

O procedimento Busca Tabu (*Tabu Search - TS*), (Glover e Laguna, 1997) implementado começa sua execução partindo de uma solução inicial gerada pelo procedimento GRASP-VND. A cada iteração todos os vizinhos desta solução são avaliados, sendo escolhido o melhor deles que não seja tabu ou, se tabu, satisfaça a condição de aspiração, no caso, de o vizinho gerar um valor melhor que o da melhor solução até então. A exploração da vizinhança é feita tendo por base as estruturas de vizinhança N^T e N^R, alternando-as a cada iteração, isto é, na primeira iteração explora-se o

melhor vizinho com movimentos de troca, na segunda iteração com movimentos de realocação e assim por diante. O objetivo de se verificar se o movimento é tabu ou não, é evitar ciclos de soluções recentemente visitadas.

Quando um vizinho $v' \in N^T(v) \cup N^R(v)$ é escolhido, ele se torna a solução corrente, independentemente de ele ser melhor ou pior que a solução corrente e um atributo que caracterize esta solução é inserido em uma lista tabu T, proibindo o retorno a ele durante um prazo. Dada a troca entre uma tarefa i e uma tarefa j ($i < j$), o atributo que é inserido na lista tabu é a subsequência (tarefa i, tarefa sucessora de i da solução v). No caso de realocação de uma tarefa i para frente (realocação progressiva), o movimento tabu é é a subsequência (tarefa i, sucessor da tarefa i na solução v); caso a realocação seja para trás (realocação regressiva) o movimento tabu é é a subsequência (tarefa antecessora a i na solução v, tarefa i). A lista tabu é limitada a $|T|$ elementos. Quando ela está completa, o elemento mais antigo é retirado da lista, e aquele recentemente obtido é inserido, tal como em uma fila (*FIFO*).

Para mostrar o funcionamento da lista tabu, considere a sequência $v = \{5, 3, 2, 1, 4\}$ e um movimento envolvendo a troca da tarefa 5 com a tarefa 1. O resultado é a sequência vizinha $v' = \{1, 3, 2, 5, 4\}$, sendo na lista tabu T incluída a subsequência formada por (tarefa 5, tarefa subsequente a 5 na sequência v), isto é, $T \leftarrow T \cup (5, 3)$.

Já para o movimento de realocação, considere primeiro a realocação da tarefa 5 para depois da tarefa 1, resultando a sequência vizinha $v' = \{3, 2, 1, 5, 4\}$, sendo adicionada na lista tabu T a subsequência formada por (tarefa 5, tarefa subsequente a 5 na sequência v), ou seja, $T \leftarrow T \cup (5, 3)$. Agora considere que o movimento de realocação seja realocar a tarefa 1 para antes da tarefa 5, que resultará na solução vizinha $v' = \{1, 5, 3, 2, 4\}$, sendo que o movimento tabu armazenado, neste caso, será a subsequência formada por (tarefa antecessora a 1, tarefa 1 na sequência v), isto é, $T \leftarrow T \cup (2, 1)$.

Sempre que há melhora na solução global, é aplicado o procedimento VND_2 descrito na Subseção 4.3.4. Caso ele seja bem sucedido, a lista tabu é inicializada

novamente, já que se muda a região do espaço de busca onde a exploração está sendo feita. O procedimento Busca Tabu é encerrado quando forem realizadas $TSmax$ iterações sem melhora na solução global.

O pseudocódigo da Busca Tabu está descrito na Figura 4.13. A atualização do Grupo Elite (GE), linhas 11 e 18 da Figura 4.13, é feita de acordo com a Subseção 4.3.6.

Procedimento `TS(`v, $|T|$, *divElite*, *GE*, *TSmax*, *MRDmax*`)`

```
1  início
2      v* ← v                                              // Melhor Solução
3      Iter ← 0
4      MelhorIter ← 0              // Iteração mais recente que forneceu v*
5      T ← ∅                                                // Lista Tabu
6      GE ← ∅                                               // Grupo Elite
7      enquanto (Iter − MelhorIter ≤ TSmax) faça
8          Iter ← Iter + 1
9          Seja v' ← v ⊕ m o melhor elemento de V ⊆ N(v) (N = N^T se Iter for
           ímpar e N = N^R se Iter for par) tal que o movimento m não seja tabu
           (m ∉ T) ou, se tabu, v' atenda a condição de aspiração f(v') < f(v*)
10         Atualize a Lista Tabu T
11         Atualize o Grupo Elite GE
12         v ← v'
13         se f(v) < f(v*) então
14             v1 ← VND2(v, MRDmax)
15             se (f(v1) < f(v)) então
16                 v ← v1
17                 T ← ∅
18                 Atualize o Grupo Elite GE
19             fim
20             v* ← v
21             MelhorIter ← Iter
22         fim
23     fim
24     v ← v*
25     retorna v
26 fim
```

Figura 4.13: Procedimento Busca Tabu aplicado ao PSUMAA

4.3.6 Reconexão por Caminhos

Reconexão por Caminhos (*Path Relinking* - PR), proposto por Glover (1996), é uma estratégia de intensificação normalmente utilizada como pós-otimização de uma solução, ou refinamento de ótimos locais obtidos durante a busca. Dado um par de soluções, o objetivo da técnica é partir de uma delas, dita solução base, e chegar à outra, dita solução guia, por meio da adição na solução base de atributos da solução guia.

No algoritmo proposto, a Reconexão por Caminhos é utilizada como ferramenta de pós-refinamento aplicada após a execução da Busca Tabu (BT). Para tanto, durante a exploração do espaço de busca pelo procedimento BT é gerado um conjunto de soluções, conhecido como Grupo Elite (GE), que é utilizado pelo procedimento PR. Para fazer parte desse grupo, cada solução candidata deve satisfazer a um dos dois seguintes critérios:

1. ser melhor que a melhor das $|GE|$ soluções do grupo elite;
2. ser melhor que a pior das $|GE|$ soluções do grupo elite e se diferenciar delas de determinado percentual dos atributos, dado por *divElite*.

O atributo considerado é o par de tarefas i e j consecutivas. Assim, por exemplo, dadas as sequências $v_1 = \{1, 4, 3, 2, 5, 6\}$ e $v_2 = \{5, 3, 2, 1, 4, 6\}$, elas têm 40% de atributos iguais, a saber, as subsequências (1, 4) e (3, 2). O objetivo desta estratégia é evitar a inclusão de soluções similares no grupo elite. Estando o grupo elite já formado, quando uma solução entra, a de pior avaliação sai.

O procedimento PR foi implementado como segue. Para cada par de soluções elite, caminha-se de forma bidirecional, isto é, tanto da pior solução para a melhor (dita Reconexão por Caminhos Progressiva - *Forward Path Relinking*), quanto da melhor para a pior solução (dita Reconexão por Caminhos Regressiva - *Backward Path Relinking*).

Determinadas as soluções base e guia, o procedimento é executado inserindo-se gradativamente um atributo da solução guia na solução base. Por exemplo, dada

a solução guia = {5, 3, 2, 1, 4, 6}, a cada iteração insere-se um par de tarefas consecutivas desta solução, a saber, (5, 3), (3, 2), (2, 1), (1, 4), (4, 6), na solução base. Cada par é inserido na ordem em que aparece na solução guia. Após cada inserção, as tarefas da solução base que foram substituídas saem da posição que ocupavam e assumem as posições daquelas que entraram. A solução base sofre, então, uma busca local, no caso, uma descida randômica usando os movimentos de troca e realocação, tal como descrito na Subseção 4.3.4 onde se fixam os atributos da solução guia que já foram incorporados à solução base. A cada iteração, a solução base recebe todos os atributos ainda não incorporados da solução guia, um de cada vez, gerando uma nova sequência para cada atributo específico da solução guia. Ao final da iteração, o atributo inserido que produziu a melhor solução após a busca local é incorporado em definitivo à solução base, se tornando fixo. O procedimento se encerra quando a solução guia é alcançada, isto é, quando a solução base passa a ter todos os atributos da solução guia.

Para ilustrar o funcionamento do procedimento, é mostrado, a seguir, um exemplo no qual a solução guia é a sequência {5, 3, 2, 1, 4, 6} e a solução base é {2, 6, 5, 1, 4, 3}.

Passo 1: O objetivo neste passo é avaliar a inserção, na solução base, de cada um dos 5 pares de subsequências existentes da solução guia, {(5, 3), (3, 2), (2, 1), (1, 4), (4, 6)}, na ordem em que aparecem e verificar qual inserção traz o melhor resultado segundo a função de avaliação. Para tanto, procede-se como segue:

Passo 1.1) Inserir as tarefas 5 e 3 do par (5, 3) nas duas primeiras posições da solução base. Para tanto, a tarefa 5 ocupará a primeira posição da solução base, onde está a tarefa 2. Esta, por sua vez, sairá da primeira posição e irá para a terceira posição, onde está a tarefa 5, resultando na sequência {5, 6, 2, 1, 4, 3}. A seguir, é inserida a tarefa 3 na segunda posição da sequência base. Como nesta posição está a tarefa 6, esta se deslocará para a última posição, onde está localizada a tarefa 3. Após esta inserção obtém-se a sequência {5, 3, 2, 1, 4, 6}.

Passo 1.2) Inserir as tarefas 3 e 2 do par (3, 2) na segunda e terceira posição, respectivamente, da solução base. Para tanto, a tarefa 3 passará a ocupar a segunda

posição da solução base, onde está a tarefa 6. Esta, por sua vez, sairá da segunda posição e irá para a última, onde está a tarefa 3, resultando na sequência {2, 3, 5, 1, 4, 6}. A seguir, é inserida a tarefa 2 na terceira posição da sequência base. Como nesta posição está a tarefa 5, esta se deslocará para a primeira posição, onde está localizada a tarefa 2. Após esta inserção obtém-se {5, 3, 2, 1, 4, 6}.

Passo 1.3) Inserir as tarefas 2 e 1 da subsequência (2, 1) na terceira e quarta posição, respectivamente, da solução base e prosseguir como anteriormente.

Passo 1.4) Inserir as tarefas 1 e 4 da subsequência (1, 4) na quarta e quinta posição, respectivamente, da solução base e prosseguir como nos passos 1.1 e 1.2.

Passo 1.5) Inserir as tarefas 4 e 6 da subsequência (4, 6) na quinta e sexta posição, respectivamente, da solução base e prosseguir como nos passos 1.1 e 1.2.

Duas estratégias de busca local foram consideradas. Na primeira, a cada inserção faz-se uma busca local com o procedimento VND_3 descrito adiante e a inserção que produzir o melhor resultado para a função de avaliação, dentre os passos 1.1 a 1.5, é realizado. Na segunda estratégia, cada inserção é avaliada e apenas aquela que melhorara valor da função de avaliação é submetida a refinamento pelo procedimento VND_3. Este procedimento consiste em uma Descida Randômica usando movimentos de troca seguida de Descida Completa com movimentos de realocação. Neste procedimento, sempre que a Descida Completa com movimentos de realocação produzir uma solução de melhora, retorna-se à Descida Randômica com movimentos de troca. Durante a aplicação do VND_3 é proibida qualquer movimentação das tarefas da solução guia que foram inseridas. Por exemplo, se a sequência do Passo 1 resultar em melhor valor para a função de avaliação, as tarefas 5 e 3 não podem movimentar-se. Assim, ter-se-á inserido na solução base um atributo da solução guia. A inserção dos demais atributos segue um procedimento semelhante, adicionando-se o fato de que durante a busca local, não somente o par recentemente inserido é fixado, mas todos os demais pares de subsequências da solução guia já inseridos também permanecem fixos.

Toda vez que o procedimento PR gera uma solução melhor que a melhor

do grupo elite, a inserção de atributos é interrompida, o grupo elite é atualizado e o processo é reiniciado com a aplicação da Reconexão por Caminhos envolvendo a melhor solução e as demais do Grupo Elite. Se encerradas as inserções dos atributos, o procedimento tiver gerado uma solução melhor que a pior, satisfazendo aos critérios de diversidade estabelecidos anteriormente, o grupo elite é atualizado com a saída de seu pior elemento. Neste caso, também aplica-se a Reconexão por Caminhos envolvendo a solução alterada e as demais do Grupo Elite. O procedimento termina após a Reconexão ser aplicada a todos os pares de solução elite e não houver alteração na melhor solução. Neste caso, retorna-se a melhor solução encontrada, bem como o conjunto elite.

O pseudocódigo do método de Reconexão por Caminhos (PR) está descrito na Figura 4.14.

Procedimento `PR(`*v, divElite, GE*`)`

```
1  início
2      f* ← ∞
3      P ← ∅          // Lista que armazena soluções usadas no processo
4      enquanto (Critério de parada não atingido) faça
5          Corrente ← Melhor solução do grupo elite
6          Guia ← Solução escolhida aleatoriamente de GE, que não esteja
           presente em P
7          P ← P∪ {Guia}
8          enquanto (Corrente ≠ Guia) faça
9              melhorAresta ← ∅
10             melhorValor ← ∞
11             para j = 1 até total de arestas da solução guia faça
12                 v ← Corrente∪ {aresta j da solução guia}
13                 v' ← BuscaLocal(v)
14                 se (f(v') < melhorValor) então
15                     melhorAresta ← j
16                     melhorValor ← f(v')
17                     se (melhorValor < f*) então
18                         v* ← v'
19                         melhorValor ← f*
20                         Corrente ← Corrente∪ {aresta melhorAresta da
                           solução guia}
21                     fim
22                 fim
23             fim
24             v ← Corrente∪ {aresta melhorAresta da solução guia}
25         fim
26         se (melhorValor é melhor que a pior solução elite e a solução
           associada diferencia das demais soluções do grupo elite em divElite%)
           então
27             Atualize o grupo elite inserindo a solução corrente no grupo elite e
               elimine a pior solução do grupo elite
28         fim
29     fim
30     v ← v*
31     retorna v e GE
32 fim
```

Figura 4.14: Procedimento PR aplicado ao PSUMAA

4.3.7 Algoritmo GTSPR

O algoritmo proposto, denominado GTSPR, possui três fases e combina características dos procedimentos GRASP, *Variable Neighborhood Descent* (VND), Busca Tabu (*Tabu Search*) e Reconexão por Caminhos (*Path Relinking*). Seu pseudocódigo é apresentado na Figura 4.15.

Algoritmo 7: GTSPR(v, γ, $GRASPmax$, $MRDmax$)

```
1 início
2 |  v_0 ← GRASP-VND(v, γ, GRASPmax, MRDmax)
3 |  v_1 ← TS(v_0, |T|, TSmax, divElite, GE, MRDmax)
4 |  v ← PR(GE, divElite, MRDmax)
5 |  retorna v
6 fim
```

Figura 4.15: Algoritmo GTSPR

Capítulo 5

Resultados Computacionais

Neste capítulo são apresentados, na seção 5.1, os problemas-teste utilizados para avaliar os modelos exato e heurístico desenvolvidos no capítulo 4. A calibração e os parâmetros dos métodos heurísticos são apresentados na seção 5.3. Por fim, os resultados computacionais obtidos com a aplicação dos modelos exato e heurístico são apresentados e discutidos, respectivamente, nas seções 5.2 e 5.4.

5.1 Problemas-teste

Para testar o algoritmo proposto foram utilizados os mesmos problemas-teste usados em Gomes Jr. *et al.* (2007). Essa base de dados envolve número de tarefas igual a 8, 9, 10, 11, 12, 15, 20, 25, 30, 40, 50 e 75, e foram geradas por estes autores utilizando os mesmos parâmetros adotados em Wan e Yen (2002), Liaw (1999) e Mazzini e Armentano (2001).

Nessa base de dados os tempos de processamento P_i das tarefas são números inteiros entre 1 e 100, enquanto os centros das janelas de entrega estão distribuídos no intervalo $[(1 - TF - RDD/2)/MS, (1 - TF + RDD/2)/MS]$, sendo MS o tempo total de processamento de todas as tarefas, TF o fator de atraso e RDD a variação relativa da janela de entrega.

Os valores de TF são 0,1; 0,2; 0,3 e 0,4 enquanto que para RDD são 0,8; 1,0 e 1,2. Os tamanhos das janelas de entrega são inteiros no intervalo $[1, MS/n]$. Os custos por atraso da produção (α_i) são números inteiros no intervalo [20, 100],

enquanto que os custos por antecipação da produção (β_i) são k vezes o custo por atraso da mesma tarefa, sendo k um número real aleatório no intervalo [0, 1]. Essa última consideração é feita tendo em vista que o atraso da produção é menos desejável do que a sua antecipação na maioria dos casos reais.

Os tempos de preparação da máquina (S_{ij}) são números inteiros no intervalo [0, 50] e são simétricos, ou seja, $S_{ij} = S_{ji}$. Todas as distribuições são uniformes. Considerando-se os diferentes valores de *TF* e *RDD*, há 144 problemas-teste no total, sendo 12 para cada número de tarefas.

5.2 Resultados do PLIM

Para realização dos experimentos computacionais do modelo matemático apresentado na Seção 4.1, o modelo foi implementado usando a ferramenta de modelagem AMPL e resolvido pelo software de otimização CPLEX, versão 9.0 da ILOG. Foram resolvidos os problemas-teste de dimensões menores, no caso, de 8 a 12 tarefas, uma vez que acima deste número de tarefas o otimizador parava por estouro de memória. Os resultados encontrados são apresentados na Tabela 5.1. Nesta tabela, a primeira coluna apresenta o número de tarefas do conjunto de problemas-teste, a segunda coluna apresenta o tempo computacional médio gasto para se chegar à solução ótima e a última coluna apresenta o percentual de soluções ótimas encontradas dentro dos 12 problemas-teste relativos a cada número de tarefas. Considerou-se como critério de parada, para obtenção de uma solução ótima, um tempo limite de 3600 segundos.

Tabela 5.1: Resultados do PLIM

Número de tarefas	Tempo médio de processamento (s)	Percentual de Soluções Ótimas Encontradas (%)
8	1,14	100,00
9	24,13	100,00
10	55,29	100,00
11	1.519,30	75,00
12	1.869,55	58,33

5.3 Calibração e Parâmetros do Método Heurístico

O algoritmo heurístico proposto foi desenvolvido na linguagem C++, utilizando o ambiente *NetBeans* 6.5 e o compilador *MinGW* (GCC 4.2.1). Todos os experimentos foram realizados em um computador *Pentium Core 2 Quad* (Q6600) 2,4 GHz com 4 GB de memória RAM e sistema operacional *Windows XP*. Apesar de o processador deste equipamento possuir 4 núcleos, o algoritmo desenvolvido não foi otimizado para multiprocessamento. Cada problema-teste foi executado 30 vezes.

Os parâmetros do algoritmo foram calibrados em uma bateria inicial de testes. A exceção foi o conjunto Γ de valores γ da fase de construção GRASP, que foi o mesmo adotado por Gomes Jr. *et al.* (2007). Todos os testes de calibração foram feitos usando-se um problema-teste de porte médio, de 25 tarefas, com 10 execuções. Calibrado um parâmetro, este era fixado na calibração dos parâmetros subsequentes.

O primeiro parâmetro a ser calibrado foi o número máximo de iterações GRASP, $GRASPmax$. O gráfico da Figura 5.1 apresenta os valores da função objetivo (Fo), para o problema-teste adotado, variando-se o valor de $GRASPmax$ de 1 a 30. É possível observar que para $GRASPmax$ superior a 20, praticamente não há alteração do valor de Fo.

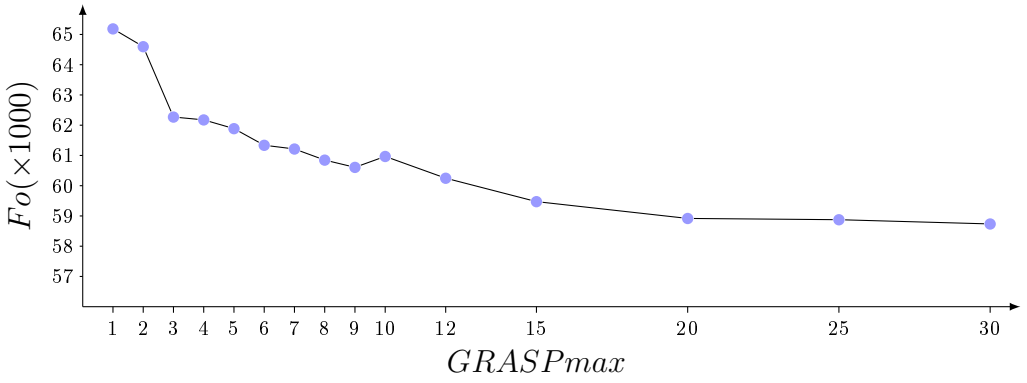

Figura 5.1: Calibração do parâmetro $GRASPmax$ em função de Fo

Já o gráfico da Figura 5.2 apresenta o tempo computacional, em segundos, gasto para resolver o problema. É possível observar que o tempo cresce linearmente ao se variar o valor de $GRASPmax$.

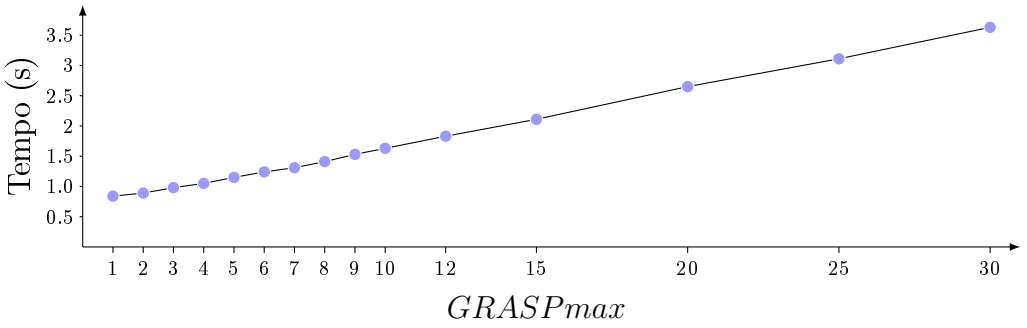

Figura 5.2: Calibração do parâmetro $GRASPmax$ em função do tempo

Ao selecionar o valor de $GRASPmax$, foram adotados dois valores, o primeiro, chamado de principal, foi adotado o valor $GRASPmax = 20$, pois a partir deste ponto o valor de Fo permanece, praticamente, constante. Para o segundo valor, chamado de secundário, deu-se prioridade ao tempo de resposta do método com uma boa solução, sendo o valor adotado $GRASPmax = 6$, pois este apresenta um bom valor para Fo, com um baixo tempo computacional. Para $GRASPmax = 20$, por exemplo, obtém-se uma melhora na solução inferior a 5% mas, em contrapartida, dobra-se o esforço computacional.

O próximo parâmetro calibrado foi o número máximo de iterações sem melhora dos procedimentos de Descida Randômica ($MRDmax$). Neste teste de calibração, os valores de $MRDmax$ foram variados em função de n, onde n é o número de tarefas do problema, de $1 \times n$ a $10 \times n$. A Figura 5.3, apresenta os testes em função de Fo e a Figura 5.4 em função do tempo. O valor adotado foi $MRDmax = 7 \times n$, visto que a partir deste ponto o valor de Fo permanece, praticamente, constante.

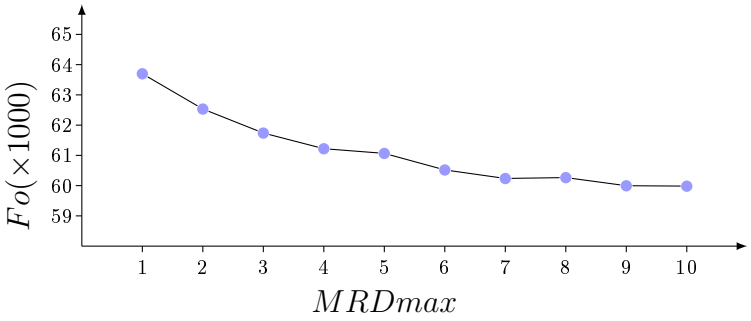

Figura 5.3: Calibração do parâmetro $MRDmax$ em função de Fo

Assim como os parâmetros do GRASP, foram calibrados os parâmetros da

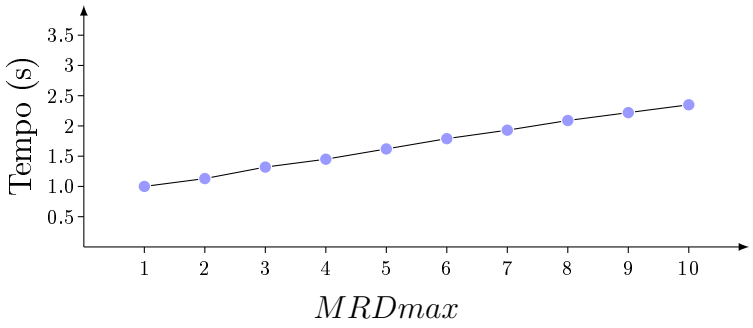

Figura 5.4: Calibração do parâmetro $MRDmax$ em função do tempo

Busca Tabu, $TSmax$, número máximo de iterações sem melhora da Busca Tabu, e $|T|$, tamanho máximo da Lista Tabu.

Os gráficos das Figuras 5.5 e 5.6 apresentam os testes de calibração do parâmetro $TSmax$, em função do valor de Fo e do tempo, respectivamente. É possível observar que o valor de Fo reduz gradativamente para valores de $TSmax$ entre $1 \times n$ e $4 \times n$; a partir deste ponto, Fo praticamente permanece constante. Assim como em $GRASPmax$ foram adotados dois valores, o principal foi $TSmax = 4 \times n$. O valor secundário selecionado para este parâmetro foi $TSmax = 2 \times n$. A justificativa deveu-se ao tempo computacional requerido por $TSmax = 4 \times n$ que produziu solução de melhor qualidade, mas requereu o dobro de tempo daquele demandado por $TSmax = 2 \times n$ para um ganho de cerca de 2%.

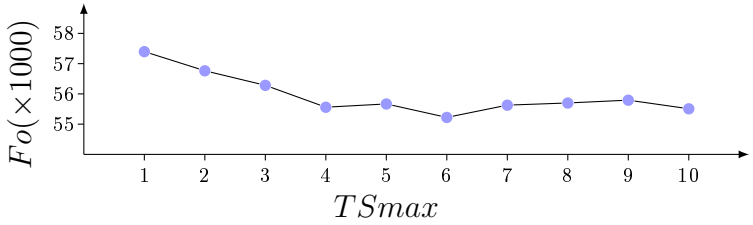

Figura 5.5: Calibração do parâmetro $TSmax$ em função de Fo

Outro parâmetro do método Busca Tabu calibrado foi o tamanho da lista tabu, $|T|$. Os gráficos das Figuras 5.7 e 5.8 apresentam os testes de calibração do mesmo.

Pelo gráfico da Figura 5.7 é possível observar que para valores de $|T|$ entre $2 \times n$ e $8 \times n$, o valor de Fo é praticamente constante. O gráfico mostra, ainda,

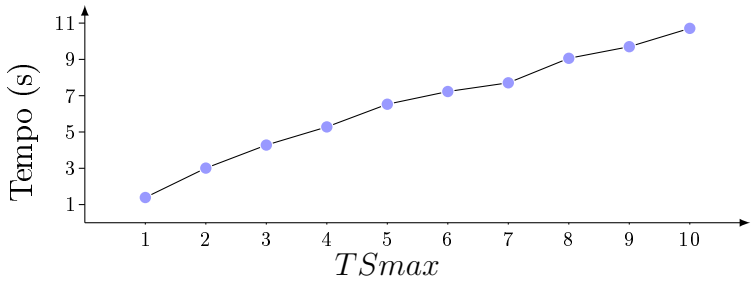

Figura 5.6: Calibração do parâmetro $TSmax$ em função do tempo

que para valores muito grandes de $|T|$, no caso, acima de $8 \times n$, os movimentos permitidos ficam restritos porque o valor da função objetivo se degrada. De acordo com o gráfico da Figura 5.8 pode-se observar que o tempo de processamento não sofre influência significativa do tamanho da lista tabu. Assim, adotou-se $|T| = 2 \times n$.

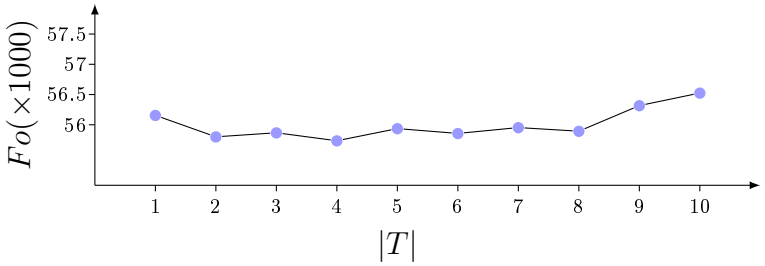

Figura 5.7: Calibração do parâmetro $|T|$ em função de Fo

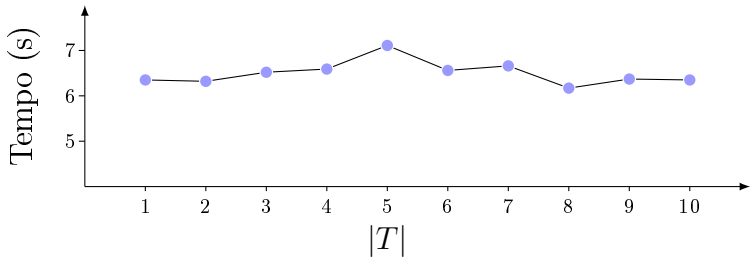

Figura 5.8: Calibração do parâmetro $|T|$ em função do tempo

Por fim, foram calibrados os parâmetros do método Reconexão por Caminhos. O tamanho máximo do Grupo Elite, $|GE|$ e o percentual mínimo de diversificação entre os membros do Grupo Elite, $divElite$.

Os gráficos das Figuras 5.9 e 5.10 mostram os valores de Fo e o tempo de processamento, respectivamente, para a calibração do parâmetro $|GE|$. Levando-se em consideração estes gráficos, foi adotado $|GE| = 5$.

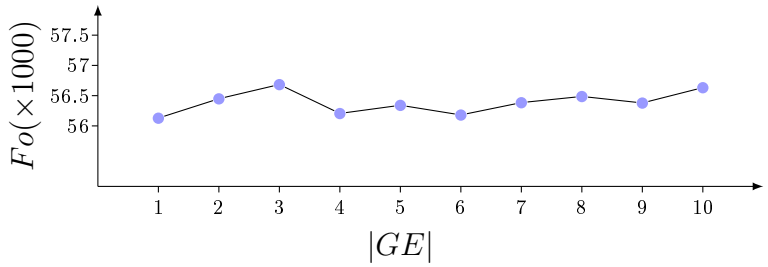

Figura 5.9: Calibração do parâmetro $|GE|$ em função de Fo

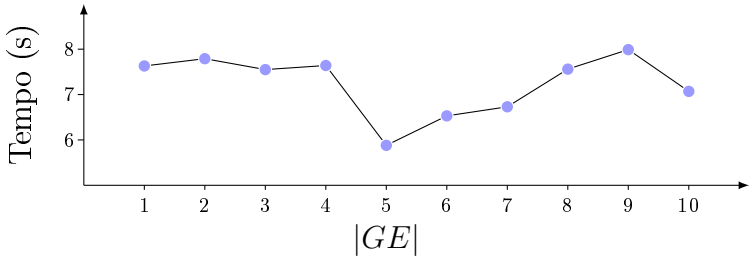

Figura 5.10: Calibração do parâmetro $|GE|$ em função do tempo

Os gráficos das Figuras 5.11 e 5.12 detalham os valores de Fo e o tempo de processamento, respectivamente, para a calibração do parâmetro *divElite*. Por análise semelhante a dos casos anteriores foi adotado $divElite = 40\%$.

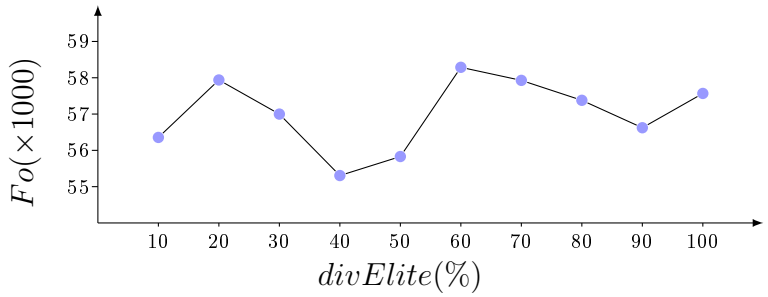

Figura 5.11: Calibração do parâmetro *divElite* em função de Fo

Os parâmetros adotados a partir dos testes de calibração estão sintetizados na Tabela 5.2, onde n representa o número de tarefas.

Nesta tabela, a segunda coluna apresenta os valores que geraram os melhores resultados nos testes de calibração. Já a terceira coluna apresenta os valores para os parâmetros *GRASPmax TSmax* que obtiveram resultados satisfatórios, porém com um esforço computacional bem inferior.

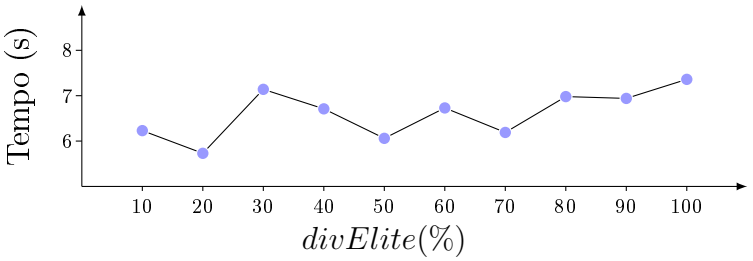

Figura 5.12: Calibração do parâmetro *divElite* em função do tempo

Tabela 5.2: Parâmetros do Algoritmo GTSPR - Solução Completa

Parâmetro	Valores Principais	Valores Secundários
Γ	{0; 0,02; 0,04; 0,12; 0,14}	{0; 0,02; 0,04; 0,12; 0,14}
GRASPmax	20	6
MRDmax	$7 \times n$	$7 \times n$
TSmax	$4 \times n$	$2 \times n$
$\|T\|$	$2 \times n$	$2 \times n$
$\|GE\|$	5	5
divElite	0,4	0,4

5.4 Resultados do Algoritmo Heurístico

Os resultados encontrados nos experimentos computacionais estão sintetizados nas Tabelas 5.3 e 5.4, onde a primeira coluna indica o número de tarefas do problema, e a cada conjunto de três colunas é apresentada a contribuição de cada fase do algoritmo GTSPR no esforço de minimização da função de avaliação f. A coluna "Fase GRASP-VND" diz respeito à fase de obtenção da solução inicial, a coluna "Fase TS" à fase de refinamento por Busca Tabu e a coluna "Fase PR", à fase de pós-otimização por Reconexão por Caminhos. As colunas "Tempo" dizem respeito ao tempo médio de processamento acumulado, em segundos.

Para cada fase do algoritmo são aplicadas duas medidas de desempenho, dadas pelas equações (5.1) e (5.2). Nestas equações, para cada problema-teste i do grupo, $f_i^{GTSPR^*}$ representa o melhor valor encontrado pelo algoritmo proposto, $f_i^{GJr^*}$ é o melhor valor encontrado por Gomes Jr. *et al.* (2007), f_i^* é o valor da melhor solução conhecida e $\bar{f}_i^{GTSPR}$ é o valor médio encontrado pelo algoritmo proposto nas diversas execuções para o i-ésimo problema-teste.

$$imp_i^{best} = \frac{f_i^{GJr^*} - f_i^{GTSPR^*}}{f_i^{GJr^*}} \tag{5.1}$$

$$gap_i^{avg} = \frac{\bar{f}_i^{GTSPR} - f_i^*}{f_i^*} \tag{5.2}$$

Na primeira medida de desempenho, verifica-se o quanto o algoritmo proposto superou o melhor resultado de Gomes Jr. *et al.* (2007). Para tanto, para cada problema-teste, aplica-se a equação (5.1) e apresenta-se nas Tabelas 5.3 e 5.4, para cada fase do algoritmo, a média percentual desses valores para cada grupo de problemas-teste com mesmo número de tarefas.

Na segunda medida de desempenho, é utilizada a equação (5.2) para calcular o desvio das soluções médias do algoritmo em relação às melhores soluções existentes. Nesta medida, quanto menor o desvio (*gap*), menor a variabilidade das soluções finais e mais robusto é o algoritmo. Nas Tabelas 5.3 e 5.4 são apresentadas, para cada fase do algoritmo, as médias percentuais dos valores dos desvios médios para cada grupo de problemas-teste.

A Tabela 5.3 apresenta os resultados adotando-se os parâmetros principais dos testes de calibração utilizando-se a primeira estratégia da Reconexão por Caminhos, isto é, aplicar a busca local VND_3 a cada inserção e escolher a inserção que produzir o melhor resultado.

Pela Tabela 5.3 é possível observar que para problemas com até 20 tarefas, apenas a fase GRASP-VND do algoritmo é suficiente para encontrar as melhores soluções, com um desvio médio de até 1,94%. Nesses problemas-teste, o desvio médio é reduzido na fase TS, ficando em até 0,99%, enquanto que essa variabilidade das soluções finais é reduzida para um máximo de 0,64%, na média, com a fase PR.

Para problemas com 25 a 50 tarefas, observa-se que a fase GRASP-VND retorna resultados piores que os de Gomes Jr. *et al.* (2007). Entretanto, a fase TS consegue reduzir essa diferença e melhorar os resultados destes autores para todos os problemas-teste desta classe. Com a fase PR, as melhores soluções de Gomes Jr. *et al.* (2007) para os problemas-teste de 25 e 30 tarefas são encontradas, enquanto que

Tabela 5.3: Comparação de resultados de cada fase do algoritmo GTSPR × Gomes Jr. *et al.* (2007) × CPLEX (Parâmetros Principais)

#	Fase GRASP-VND			Fase TS			Fase PR			Gomes Jr.	CPLEX
Tar.	$\overline{imp}^{best}$	$\overline{gap}^{avg}$	tempo[1]	$\overline{imp}^{best}$	$\overline{gap}^{avg}$	tempo[1]	$\overline{imp}^{best}$	$\overline{gap}^{avg}$	Tempo[1]	Tempo[2]	Tempo[1]
	%	%	(s)	%	%	(s)	%	%	(s)	(s)	(s)
8	0,00	0,00	0,03	0,00	0,00	0,04	0,00	0,00	0,06	0,04	1,14
9	0,00	0,00	0,05	0,00	0,00	0,06	0,00	0,00	0,09	0,07	24,13
10	0,00	0,00	0,07	0,00	0,00	0,09	0,00	0,00	0,15	0,11	55,29
11	0,00	0,13	0,10	0,00	0,00	0,14	0,00	0,00	0,25	0,20	1519,30
12	0,00	0,07	0,14	0,00	0,01	0,20	0,00	0,00	0,37	0,29	1869,55
15	0,00	1,15	0,33	0,00	0,63	0,54	0,00	0,47	1,13	0,94	-
20	0,00	1,94	1,06	0,00	0,99	2,03	0,00	0,64	4,93	4,35	-
25	-0,57	2,94	2,59	0,00	1,46	6,13	0,00	1,09	14,90	13,29	-
30	-0,22	4,54	6,24	0,00	2,13	16,78	0,00	1,68	39,93	40,07	-
40	-1,09	7,26	20,77	0,15	3,81	70,20	0,18	3,24	193,61	155,79	-
50	-1,15	10,08	57,36	0,51	5,45	258,63	0,74	4,61	630,77	492,28	-
75	1,38	9,71	399,09	5,32	7,14	2460,59	5,75	6,56	6308,74	1368,08	-
Avg	-0,14	3,17	-	0,50	1,80	-	0,56	1,52	-	-	-

[1] Tempo de CPU, em um PC Intel Core 2 Quad (Q6600) 2,40 GHz com 4 GB de RAM

[2] Tempo de CPU, em um PC Athlon XP 64 Bits 3000+ (aprox. 2 GHz) com 1 GB de RAM

para os demais problemas-teste, o algoritmo proposto consegue superar os resultados destes autores na média em até 0,74%. Por fim, os problemas-teste envolvendo 75 tarefas apresentaram melhoras significativas já na primeira fase de 1,38%, passando a 5,75% na terceira fase. Vale ressaltar que para os problemas-teste com 75 tarefas, Gomes Jr. *et al.* (2007) utilizaram parâmetros de calibração diferentes, em seus testes, como meio de reduzir o esforço computacional. Com relação ao desvio médio para esta classe de problemas, a variabilidade das soluções finais reduziu de 10,08% na fase GRASP-VND para 7,14% na fase TS e para 6,56% na fase PR. Observa-se que no algoritmo de Gomes Jr. *et al.* (2007), os desvios médios finais foram de até 18,53%. Este último percentual foi ajustado de acordo com as novas melhores soluções encontradas.

Em termos de tempos computacionais, uma comparação direta de desempenho não é possível, uma vez que as máquinas utilizadas para teste foram diferentes. Contudo, uma comparação mais justa é feita mais adiante, por meio da Tabela 5.5, que apresenta um comparativo de desempenho entre os algoritmos com um ajuste no tempo de processamento do algoritmo de Gomes Jr. *et al.* (2007).

Ainda sobre os tempos computacionais, pode-se verificar que o tempo gasto pelo GTSPR, para os problemas até 12 tarefas foi bem inferior ao tempo gasto pelo CPLEX, apresentado na última coluna, para encontrar a solução ótima. Vale ressaltar que não houve desvio na solução final apresentada pelo algoritmo proposto.

Com a finalidade de melhorar os tempos computacionais exigidos pelo algoritmo proposto, uma segunda bateria de testes foi executada, desta vez, utilizando os parâmetros secundários dos testes de calibração e a segunda estratégia da Reconexão por Caminhos, isto é, a busca local VND_3 é aplicada somente se a inserção de um arco produzir uma melhora. A Tabela 5.4 apresenta estes resultados.

Tabela 5.4: Comparação de resultados de cada fase do algoritmo GTSPR × Gomes Jr. *et al.* (2007) × CPLEX (Parâmetros Secundários)

#	Fase GRASP-VND			Fase TS			Fase PR			Gomes Jr.	CPLEX
Tar.	$\overline{imp}^{best}$	$\overline{gap}^{avg}$	tempo$^{(1)}$	$\overline{imp}^{best}$	$\overline{gap}^{avg}$	tempo$^{(1)}$	$\overline{imp}^{best}$	$\overline{gap}^{avg}$	Tempo$^{(1)}$	Tempo$^{(2)}$	Tempo$^{(1)}$
	%	%	(s)	%	%	(s)	%	%	(s)	(s)	(s)
8	0,00	0,00	0,01	0,00	0,00	0,02	0,00	0,00	0,03	0,04	1,14
9	0,00	0,21	0,02	0,00	0,00	0,03	0,00	0,00	0,03	0,07	24,13
10	0,00	0,35	0,02	0,00	0,08	0,04	0,00	0,08	0,05	0,11	55,29
11	0,00	0,83	0,04	0,00	0,20	0,06	0,00	0,18	0,09	0,20	1519,30
12	0,00	0,83	0,05	0,00	0,52	0,08	0,00	0,43	0,13	0,29	1869,55
15	0,00	3,97	0,12	0,00	2,31	0,24	0,00	2,08	0,32	0,94	-
20	0,00	5,35	0,42	0,00	2,36	1,04	0,00	2,05	1,32	4,35	-
25	-0,66	6,65	1,09	0,00	3,03	3,29	0,00	2,82	4,11	13,29	-
30	-0,67	8,07	2,75	-0,05	3,62	9,83	0,00	3,48	11,63	40,07	-
40	-1,24	10,12	9,98	0,17	5,27	42,29	0,17	5,07	47,64	155,79	-
50	-2,70	13,54	29,88	0,60	7,25	165,99	0,61	7,10	181,12	492,28	-
75	0,58	16,99	256,31	4,53	9,84	1728,05	4,65	9,65	1828,88	1368,08	-
Avg	-0,39	5,58	-	0,44	2,87	-	0,45	2,75	-	-	-

Com esta segunda bateria de testes, como era de se esperar, pode-se observar que os tempos computacionais tiveram uma redução significativa, ficando bem abaixo dos tempos demandados pelo algoritmo de Gomes Jr. *et al.* (2007), mesmo sem o ajuste, com exceção dos problemas com 75 tarefas. Apesar da significativa diminuição dos tempos verifica-se que a qualidade das soluções foi pouco afetada. Para as problemas com até 20 tarefas, continuou sendo necessária apenas a primeira fase do algoritmo proposto para encontrar a melhor solução, com um desvio médio máximo de 5,35%, passando a 2,36% na segunda fase e caindo para 2,05% na ter-

ceira fase. Para os problemas com 25 ou mais tarefas, a fase TS foi suficiente para resolvê-los e melhorar os resultados, à exceção dos problemas com 30 tarefas, em que foi necessária a fase PR. As soluções finais tiveram uma melhora de até 4,65%, com desvio máximo de 9,65%, valor este ainda melhor que os 18,53% obtidos por Gomes Jr. *et al.* (2007). Observa-se que este último percentual foi ajustado de acordo com as novas melhores soluções encontradas.

Pelas Tabelas 5.3 e 5.4 observa-se que a fase PR demanda um tempo computacional elevado e sua contribuição é relativamente baixa, servindo basicamente para diminuir a variabilidade da qualidade da solução final. Observa-se, ainda, que apenas as fases GRASP-VND e TS são suficientes para resolver o problema com um esforço computacional muito menor e sem detrimento significativo da qualidade da solução final.

A Tabela 5.5 apresenta uma comparação dos tempos computacionais das duas baterias de testes do algoritmo GTSPR. A segunda coluna, GTSPR I, apresenta os tempos computacionais com os parâmetros principais, enquanto a terceira coluna, GTSPR II, os tempos com os parâmetros secundários. A quarta coluna apresenta os tempos demandados pelo algoritmo de Gomes Jr. *et al.* (2007) reduzindo-os em 30%. Este valor foi obtido baseado em testes encontrados na página http://www.tomshardware.com/charts/cpu-charts-2008-q1-2008/compare,372.html?prod[1275]=on&prod[1218]=on, que compara os tempos gastos na execução de programas como antivirus e compactadores nas máquinas envolvidas no presente trabalho. Nestes testes os programas executados em máquinas com processador PC Intel Core 2 Quad (Q6600) obtiveram tempos computacionais em média 30% inferiores aos mesmos programas executados em máquinas com processador PC Athlon XP 64 3000+.

Ao se comparar os tempos computacionais dos algoritmos, vê-se que GTSPR II demandou menor tempo de processamento para produzir soluções com qualidade melhor ou igual àquelas geradas pelo algoritmo de Gomes Jr. *et al.* (2007). A exceção ocorreu nos problemas de 75 tarefas, nos quais foi exigido um tempo duas vezes maior, mas que resultou, em contrapartida, em soluções finais melhores em

até 4,65%, na média. Além disso, já na fase GRASP-VND, GTSPR II demandou 256,31 segundos para obter soluções, em média 0,58% melhores do que o algoritmo de Gomes Jr. *et al.* (2007) em 957,66 segundos.

Tabela 5.5: Comparação dos tempos computacionais GTSPR I × GTSPR II × Gomes Jr. *et al.* (2007) corrigido

# Tarefas	GTSPR I Tempo (s)	GTSPR II Tempo (s)	Gomes Jr. *et al.* (2007) Tempo corrigido* (s)
08	0,06	0,03	0,03
09	0,09	0,03	0,05
10	0,15	0,05	0,08
11	0,25	0,09	0,14
12	0,37	0,13	0,20
15	1,13	0,32	0,66
20	4,93	1,32	3,05
25	14,90	4,16	9,30
30	39,93	11,54	28,05
40	173,58	47,64	109,05
50	630,77	182,12	344,60
75	6308,74	1828,88	957,66

(*) Tempo de CPU corrigido em 30%, com base em testes comparativos entre performance de processadores

No Anexo I são apresentados os resultados detalhados para os problemas-teste utilizados. Nas Tabelas I.1 a I.4 são mostrados os resultados para a execução GTSPR I e nas Tabelas I.5 a I.8, os resultados do GTSPR II.

Capítulo 6

Conclusões e Trabalhos Futuros

Este trabalho teve seu foco na resolução do problema de sequenciamento em uma máquina com penalidades por antecipação e atraso da produção (PSUMAA), considerando janelas de entrega e tempo de preparação da máquina dependente da sequência de produção.

Foi proposto um algoritmo de três fases, denominado GTSPR, que combina os procedimentos GRASP e Descida em Vizinhança Variável (*Variable Neighborhood Descent* - VND), para geração de solução inicial; Busca Tabu (*Tabu Search* - TS) para refinamento e Reconexão por Caminhos (*Path Relinking* - PR), como mecanismo de pós-otimização. Para cada sequência gerada pelo algoritmo são determinadas as datas ótimas de conclusão das tarefas por meio de um procedimento de tempo polinomial da literatura. Para explorar o espaço de soluções são utilizados três tipos de movimentos, baseados em troca entre tarefas, realocação de uma tarefa para outra posição e realocação de um bloco de tarefas para outra posição.

Inicialmente o algoritmo proposto foi comparado com o otimizador CPLEX aplicado a problemas-teste envolvendo até 12 tarefas. Os resultados obtidos mostraram que todas as soluções ótimas foram encontradas em tempos computacionais muito inferiores aos demandados pelo otimizador e o desvio máximo foi de 0,43%.

A seguir, o algoritmo GTSPR foi aplicado a problemas-teste envolvendo até 75 tarefas e comparado com um algoritmo da literatura. Os experimentos computacionais mostraram que, para problemas-teste de até 20 tarefas, apenas a fase

GRASP-VND do algoritmo foi suficiente para alcançar as melhores soluções existentes. Todas as demais fases do algoritmo tiveram sua contribuição para a melhora da qualidade da solução final, reduzindo sua variabilidade. Com o algoritmo proposto foi possível superar os resultados da literatura em até 5,75%, com uma variabilidade média de 6,56%, no máximo. Em termos de tempo computacional, o algoritmo proposto também foi competitivo, sendo capaz de produzir soluções melhores em menor tempo de processamento.

Como trabalhos futuros, indicam-se os seguintes aperfeiçoamentos:

1. Aplicar periodicamente a estratégia de Reconexão por Caminhos durante a exploração da busca, e não somente como mecanismo de pós-otimização;
2. Aplicar o princípio da otimalidade próxima durante a fase de construção de uma solução;
3. Estudar propriedades que possam ser aplicadas ao problema, de forma a reduzir o espaço de busca e, consequentemente, o tempo de processamento do algoritmo.

Referências Bibliográficas

Allahverdi, A.; Gupta, J. N. D. e Aldowaisan, T. (1999). A review of scheduling research involving setup considerations. *OMEGA*, v. 27, p. 219–239.

Allahverdi, Ali; Ng, C.T.; Cheng, T.C.E. e Kovalyov, Mikhail Y. (2008). A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, v. 187, n. 3, p. 985–1032.

Araújo, Francisco César Rodrigues de. (2008). Planejamento operacional de lavra com alocação dinâmica de caminhões: Abordagens exata e heurística. Dissertação de mestrado, Departamento de Engenharia de Minas/EM/UFOP, Ouro Preto.

Arenales, M.; Armentano, V.; Morabito, R. e Yanasse, H. (2007). *Pesquisa Operacional para cursos de Engenharia*. Editora Campus, Rio de Janeiro.

Baker, K. R. e Scudder, G. D. (1990). Sequencing with earliness and tardiness penalties: A review. *Operations Research*, v. 38, p. 22–36.

Bilge, Ümit; Kurtulan, Müjde e Kirac, Furkan. (2007). A tabu search algorithm for the single machine total weighted tardiness problem. *European Journal of Operational Research*, v. 176, p. 1423–1435.

Biskup, D. e Feldmann, M. (2001). Benchmarks for scheduling on a single machine against restrictive and unrestrictive common due dates. *Computers and Operations Research*, v. 28, p. 787–801.

Bustamante, L. M. (2006). Minimização do custo de antecipação e atraso para o problema de seqüenciamento de uma máquina com tempo de preparação dependente da seqüência: Aplicação em uma usina siderúrgica. Dissertação de mestrado, Programa de Pós-Graduação em Engenharia de Produção, UFMG, Belo Horizonte.

Chang, P. C. (1999). A branch and bound approach for single machine scheduling with earliness and tardiness penalties. *Computers & Mathematics with Applications*, v. 37, n. 10, p. 133–144.

Chen, Zhi-Long. (1997). Scheduling with batch setup times and earliness-tardiness penalties. *European Journal of Operational Research*, v. 96, p. 518–537.

Coleman, B. J. (1992). A simple model for optimizing the single machine early/tardy problem with sequence-dependent setups. *Production and Operation Management*, v. 1, n. 2, p. 225–228.

Costa, Felippe Pereira. (2005). Aplicações de técnicas de otimização a problemas de planejamento operacional de lavra em minas a céu aberto. Dissertação de mestrado, Departamento de Engenharia de Minas/EM/UFOP, Ouro Preto.

Du, J. e Leung, J. Y. T. (1990). Minimizing total tardiness on one machine is np-hard. *Mathematics of Operations Research*, v. 15, p. 483–495.

Feldmann, Martin e Biskup, Dirk. (2003). Single-machine scheduling for minimizing earliness and tardiness penalties by meta-heuristic approaches. *Computers and Industrial Engineering*, v. 44, p. 307–323.

Feo, T.A. e Resende, M.G.C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, v. 6, p. 109–133.

França Filho, M. F. (2007). Grasp e busca tabu aplicados a problemas de programação de tarefas em máquinas paralelas. Tese de doutorado, Departamento de Engenharia de Sistemas/UNICAMP, Campinas.

Gagné, Caroline; Price, Wilson L. e Gravel, Marc. (2001). Scheduling a single machine with sequence dependent setup times using ant colony optimization. Relatório Técnico 2001-003, Faculté des sciences de l'administration, Université Laval.

Gershon, M. (1982). A linear programming approach to mine scheduling optimization. *Proceedings of the 17th Application of computers and operations research in the mineral industry*, p. 483–493, New York.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, v. 5, p. 553–549.

Glover, F. (1996). *Computing Tools for Modeling, Optimization and Simulation: Interfaces in Computer Science and Operations Research*, Capítulo Tabu search and adaptive memory programming - Advances, applications and challenges, p. 1–75. Kluwer Academic Publishers.

Glover, F. e Kochenberger, G. (2003). *Handbook of Metaheuristics*. Kluwer Academic Publishers.

Glover, F. e Laguna, M. (1997). *Tabu Search*. Kluwer Academic Publishers, Boston.

Gomes Jr., A. C.; Carvalho, C. R. V.; Munhoz, P. L. A. e Souza, M. J. F. (2007). Um método heurístico híbrido para a resolução do problema de sequenciamento em uma máquina com penalidades por antecipação e atraso da produção. *Anais do XXXIX Simpósio Brasileiro de Pesquisa Operacional, SBPO*, p. 1649–1660, Fortaleza.

Gordon, Valery; Proth, Jean-Marie e Chu, Chengbin. (2002). A survey of the state-of-the-art of common due date assignment and scheduling research. *European Journal of Operational Research*, v. 139, p. 1–25.

Gupta, S. R. e Smith, J. S. (2006). Algorithms for single machine total tardiness scheduling with sequence dependent setups. *European Journal of Operational Research*, v. 175, p. 722–739.

Hallah, R. M. (2007). Minimizing total earliness and tardiness on a single machine using a hybrid heuristic. *Computers and Operations Research*, v. 34, p. 3126–3142.

Hassin, Refael e Shani, Mati. (2005). Machine scheduling with earliness, tardiness and non-execution penalties. *Computers and Operations Research*, v. 32, p. 683–705.

Hino, C. M.; Ronconi, D. P. e Mendes, A. B. (2005). Minimizing earliness and tardiness penalties in a single-machine problem with a common due date. *European Journal of Operational Research*, v. 160, p. 190–201.

Jackson, J. R. (1955). Scheduling a production line to minimize maximum tardiness. Relatório técnico, UCLA, Management Science Research Project.

James, R. J. W. (1997). Using tabu search to solve the common due date early/tardy machine scheduling problem. *Computers and Operations Research*, v. 24, n. 3, p. 199–208.

Kim, S. C. e Bobrowski, P. M. (1994). Impact os sequence dependent setup time on job shop scheduling performance. v. 32, n. 7, p. 1503–1520.

Lee, C. Y. e Choi, J. Y. (1995). A genetic algorithm for job sequencing problems with distinct due dates and general early-tardy penalty weights. *Computers and Operations Research*, v. 22, p. 857–869.

Li, F. (1997). Single machine earliness and tardiness scheduling. *European Journal of Operational Research*, v. 96, p. 546–558.

Liaw, C. F. (1999). A branch-and-bound algorithm for the single machine earliness and tardiness scheduling problem. *Computers and Operations Research*, v. 26, p. 679–693.

Lin, Shih-Wei; Chou, Shuo-Yan e Chen, Shih-Chieh. (2007)a. Meta-heuristic approaches for minimizing total earliness and tardiness penalties of single-machine scheduling with a common due date. *Journal of Heuristic*, v. 13, p. 151–165.

Lin, Shih-Wei; Chou, Shuo-Yan e Ying, Kuo-Ching. (2007)b. A sequential exchange approach for minimizing earliness-tardiness penalties of single-machine scheduling with a common due date. *European Journal of Operational Research*, v. 177, p. 1294–1301.

Mazzini, R. e Armentano, V. A. (2001). A heuristic for single machine scheduling with early and tardy costs. *European Journal of Operational Research*, v. 128, p. 129–146.

Mladenović, N. e Hansen, P. (1997). Variable Neighborhood Search. *Computers and Operations Research*, v. 24, p. 1097–1100.

Panwalkar, S. S.; Dudek, R. A. e Smith, M. L. (1973). Sequencing research and the industrial scheduling problem. SE, Elmaghraby, editor, *Symposium on the Theory of Scheduling and its Applications*, p. 29–38, Springer: Berlin.

Rabadi, G.; Mollaghasemi, M. e Anagnostopoulos, G. C. (2004). A branch-and-bound algorithm for the early/tardy machine scheduling problem with a common due-date and sequence-dependent setup time. *Computers and Operations Research*, v. 31, p. 1727–1751.

Rodrigues, Lásara Fabrícia. (2006). Análise comparativa de metodologias utilizadas no despacho de caminhões em minas a céu aberto. Dissertação de mestrado, Departamento de Engenharia de Produção/Escola de Engenharia/UFMG, Belo Horizonte.

Sridharan, V. e Zhou, Z. (1996). A decision theory based scheduling procedure for single-machine weighted earliness and tardiness problems. *European Journal of Operational Research*, v. 94, p. 292–301.

Tian, Z.J.; Ng, C.T. e Cheng, T.C.E. (2005). On the single machine total tardiness problem. *European Journal of Operational Research*, v. 165, p. 843–846.

Uzsoy, Reha e Velásquez, Juan Diego. (2008). Heuristics for minimizing maximum lateness on a single machine with family-dependent set-up times. *Computers and Operations Research*, v. 35, p. 2018–2033.

Ventura, José A. e Radhakrishnan, Sanjay. (2003). Single machine scheduling with symmetric earliness and tardiness penalties. *European Journal of Operational Research*, v. 144, p. 598–612.

Wan, G. e Yen, B. P. C. (2002). Tabu search for single machine scheduling with distinct due windows and weighted earliness/tardiness penalties. *European Journal of Operational Research*, v. 142, p. 271–281.

Ying, Kuo-Ching. (2008). Minimizing earliness-tardiness penalties for common due date single-machine scheduling problems by a recovering beam search algorithm. *Computers and Industrial Engineering*, v. In Press, Corrected Proof. doi: 10.1016/j.cie.2008.01.008.

Anexo I

Resultados Detalhados

I.1 Execução do GTSPR I

As Tabelas I.1 a I.4 mostram os resultados detalhados para os problemas-teste de 08 a 75 tarefas. Nestas tabelas, imp_i^{avg} é dado pela Expressão(I.1) e representa a melhora do algoritmo GTSPR I sobre o algoritmo de Gomes Jr. *et al.* (2007) em relação a solução média encontrado por cada um.

Na Expressão (I.1), $\bar{f}_i^{GJr}$ representa o valor médio obtido pelo algoritmo de Gomes Jr. *et al.* (2007) em trinta execuções e o outro membro é o mesmo da Equação (5.2).

$$imp_i^{avg} = \frac{\bar{f}_i^{GJr} - \bar{f}_i^{GTSPR}}{\bar{f}_i^{GJr}} \tag{I.1}$$

Tabela I.1: Resultados detalhados GTSPR I × Gomes Jr. *et al.* (2007) (8 a 10 tarefas)

	Melhor Valor			Valor Médio				Tempo (s)	
Inst.	Gomes Jr.	GTSPR I	$\overline{imp}^{best}$ %	Gomes Jr.	GTSPR I	$\overline{imp}^{avg}$ %	$\overline{gap}^{avg}$ %	Gomes Jr.	GTSPR I
0801	4928	4928	0,00	4928	4928	0,00	0,00	0,05	0,06
0802	2739	2739	0,00	2739	2739	0,00	0,00	0,03	0,05
0803	4074	4074	0,00	4074	4074	0,00	0,00	0,02	0,05
0804	17149	17149	0,00	17149	17149	0,00	0,00	0,05	0,07
0805	6195	6195	0,00	6195	6195	0,00	0,00	0,03	0,05
0806	4004	4004	0,00	4004	4004	0,00	0,00	0,03	0,06
0807	10689	10689	0,00	10689	10689	0,00	0,00	0,05	0,06
0808	14059	14059	0,00	14059	14059	0,00	0,00	0,04	0,06
0809	13705	13705	0,00	13751	13705	0,33	0,00	0,05	0,06
0810	19855	19855	0,00	19855	19855	0,00	0,00	0,05	0,08
0811	22774	22774	0,00	22774	22774	0,00	0,00	0,04	0,06
0812	6184	6184	0,00	6184	6184	0,00	0,00	0,03	0,05
Média			0,00			0,03	0,00	0,04	0,06
0901	11801	11801	0,00	11801	11801	0,00	0,00	0,07	0,08
0902	8561	8561	0,00	8561	8561	0,00	0,00	0,05	0,08
0903	4734	4734	0,00	4734	4734	0,00	0,00	0,05	0,09
0904	31100	31100	0,00	31124	31100	0,08	0,00	0,08	0,09
0905	2986	2986	0,00	2986	2986	0,00	0,00	0,06	0,09
0906	13845	13845	0,00	13845	13845	0,00	0,00	0,06	0,09
0907	16400	16400	0,00	16512	16400	0,68	0,00	0,08	0,09
0908	20817	20817	0,00	20817	20817	0,00	0,00	0,07	0,10
0909	14431	14431	0,00	14431	14431	0,00	0,00	0,05	0,07
0910	25664	25664	0,00	25664	25664	0,00	0,00	0,09	0,09
0911	33202	33202	0,00	33202	33202	0,00	0,00	0,08	0,10
0912	13068	13068	0,00	13068	13068	0,00	0,00	0,06	0,09
Média			0,00			0,06	0,00	0,07	0,09
1001	14411	14411	0,00	14456	14411	0,31	0,00	0,13	0,15
1002	8349	8349	0,00	8349	8349	0,00	0,00	0,11	0,15
1003	11605	11605	0,00	11605	11605	0,00	0,00	0,10	0,14
1004	12486	12486	0,00	12486	12486	0,00	0,00	0,12	0,16
1005	5679	5679	0,00	5679	5679	0,00	0,00	0,12	0,14
1006	2897	2897	0,00	2897	2897	0,00	0,00	0,09	0,13
1007	5515	5515	0,00	5515	5515	0,00	0,00	0,12	0,15
1008	9534	9534	0,00	9534	9534	0,00	0,00	0,10	0,16
1009	9461	9461	0,00	9461	9461	0,00	0,00	0,09	0,14
1010	22273	22273	0,00	22273	22273	0,00	0,00	0,12	0,16
1011	20514	20514	0,00	20514	20514	0,00	0,00	0,09	0,16
1012	45554	45554	0,00	45554	45554	0,00	0,00	0,10	0,14
Média			0,00			0,03	0,00	0,11	0,15

Tabela I.2: Resultados detalhados GTSPR I × Gomes Jr. *et al.* (2007) (11 a 15 tarefas)

	Melhor Valor			Valor Médio				Tempo (s)	
Inst.	Gomes Jr.	GTSPR I	$\overline{imp}^{best}$ %	Gomes Jr.	GTSPR I	$\overline{imp}^{avg}$ %	$\overline{gap}^{avg}$ %	Gomes Jr.	GTSPR I
1101	16530	16530	0,00	16530	16530	0,00	0,00	0,16	0,23
1102	13252	13252	0,00	13252	13252	0,00	0,00	0,18	0,28
1103	15251	15251	0,00	15251	15251	0,00	0,00	0,14	0,23
1104	17573	17573	0,00	17573	17573	0,00	0,00	0,19	0,23
1105	9347	9347	0,00	9347	9347	0,00	0,00	0,19	0,23
1106	15737	15737	0,00	15737	15737	0,00	0,00	0,22	0,22
1107	12311	12311	0,00	12311	12311	0,00	0,00	0,21	0,24
1108	17361	17361	0,00	17374	17361	0,07	0,00	0,21	0,25
1109	18202	18202	0,00	18337	18202	0,74	0,00	0,21	0,22
1110	28886	28886	0,00	28977	28886	0,31	0,00	0,23	0,27
1111	30328	30328	0,00	30328	30328	0,00	0,00	0,20	0,26
1112	76689	76689	0,00	76911	76689	0,29	0,00	0,24	0,29
Média			0,00			0,12	0,00	0,20	0,25
1201	8137	8137	0,00	8183	8137	0,56	0,00	0,26	0,37
1202	9848	9848	0,00	9858	9848	0,10	0,00	0,23	0,28
1203	8002	8002	0,00	8019	8002	0,21	0,00	0,23	0,25
1204	29466	29466	0,00	29477	29466	0,04	0,00	0,35	0,38
1205	8468	8468	0,00	8468	8468	0,00	0,00	0,27	0,34
1206	10982	10982	0,00	10982	10982	0,00	0,00	0,22	0,28
1207	26939	26939	0,00	27033	26939	0,35	0,00	0,36	0,43
1208	11335	11335	0,00	11335	11335	0,00	0,00	0,27	0,40
1209	28610	28610	0,00	28967	28610	1,23	0,00	0,31	0,37
1210	23406	23406	0,00	23420	23406	0,06	0,00	0,37	0,41
1211	23858	23858	0,00	23858	23858	0,00	0,00	0,31	0,45
1212	41983	41983	0,00	41983	41983	0,00	0,00	0,29	0,40
Média			0,00			0,21	0,00	0,29	0,37
1501	18276	18276	0,00	18276	18276	0,00	0,00	0,64	0,93
1502	19622	19622	0,00	20198	19622	2,85	0,00	1,23	1,23
1503	11505	11505	0,00	11952	11505	3,74	0,00	1,09	1,23
1504	15164	15164	0,00	15250	15164	0,56	0,00	1,00	1,16
1505	12924	12924	0,00	13141	12924	1,65	0,00	0,79	1,01
1506	9396	9396	0,00	9423	9396	0,29	0,00	0,60	0,61
1507	46544	46544	0,00	46977	46724	0,54	0,39	1,23	1,26
1508	24899	24899	0,00	26201	26220	-0,07	5,31	1,19	1,14
1509	14457	14457	0,00	14457	14457	0,00	0,00	0,56	0,90
1510	33128	33128	0,00	33556	33128	1,28	0,00	1,11	1,39
1511	42522	42522	0,00	42866	42522	0,80	0,00	1,12	1,41
1512	12793	12793	0,00	12793	12793	0,00	0,00	0,68	1,29
Média			0,00			0,97	0,47	0,94	1,13

Tabela I.3: Resultados detalhados GTSPR I × Gomes Jr. *et al.* (2007) (20 a 30 tarefas)

	Melhor Valor			Valor Médio				Tempo (s)	
Inst.	Gomes Jr.	GTSPR I	$\overline{imp}^{best}$ %	Gomes Jr.	GTSPR I	$\overline{imp}^{avg}$ %	$\overline{gap}^{avg}$ %	Gomes Jr.	GTSPR I
2001	16294	16294	0,00	16294	16294	0,00	0,00	1,95	3,69
2002	18107	18107	0,00	18660	18173	2,61	0,37	3,63	3,75
2003	20249	20249	0,00	20461	20425	0,18	0,87	3,07	3,04
2004	29941	29941	0,00	30283	29952	1,09	0,04	4,43	4,17
2005	59158	59158	0,00	59736	59535	0,34	0,64	4,60	5,85
2006	31053	31053	0,00	32456	32016	1,35	3,10	4,86	5,13
2007	40438	40438	0,00	42005	41127	2,09	1,70	4,84	5,83
2008	29186	29186	0,00	29368	29186	0,62	0,00	3,18	3,93
2009	67827	67827	0,00	69246	67827	2,05	0,00	5,61	6,49
2010	34283	34283	0,00	34656	34424	0,67	0,41	5,52	5,32
2011	54538	54538	0,00	54911	54692	0,40	0,28	5,90	5,54
2012	79599	79599	0,00	80171	79780	0,49	0,23	4,58	6,45
Média			0,00			0,99	0,64	4,35	4,93
2501	23397	23397	0,00	23480	23397	0,35	0,00	13,27	13,23
2502	48540	48540	0,00	48801	48968	-0,34	0,88	13,24	12,04
2503	18503	18503	0,00	18526	18579	-0,28	0,41	9,32	12,05
2504	25645	25645	0,00	25785	25645	0,54	0,00	13,44	13,33
2505	35865	35865	0,00	36796	35931	2,35	0,18	14,14	17,13
2506	14189	14189	0,00	14219	14189	0,21	0,00	7,85	9,94
2507	37313	37313	0,00	37857	37377	1,27	0,17	15,26	12,72
2508	44638	44638	0,00	45150	44638	1,13	0,00	13,23	15,83
2509	12839	12839	0,00	12874	12839	0,27	0,00	10,52	17,64
2510	51415	51415	0,00	57174	54850	4,06	6,68	18,25	19,11
2511	44808	44808	0,00	47610	46682	1,95	4,18	15,87	17,78
2512	34197	34197	0,00	35255	34408	2,40	0,62	15,10	18,03
Média			0,00			1,16	1,09	13,29	14,90
3001	43673	43673	0,00	44778	43760	2,27	0,20	38,91	37,30
3002	41983	41983	0,00	42111	42021	0,21	0,09	39,48	32,27
3003	21951	21951	0,00	22375	21951	1,89	0,00	24,60	32,00
3004	64943	64943	0,00	68138	66751	2,04	2,78	57,49	47,78
3005	74100	74100	0,00	76040	75170	1,14	1,44	46,12	45,98
3006	29829	29829	0,00	30832	30872	-0,13	3,50	33,51	38,17
3007	74336	74336	0,00	75378	74553	1,09	0,29	41,04	38,50
3008	69770	69770	0,00	74097	73579	0,70	5,46	53,20	47,03
3009	21335	21335	0,00	21641	21392	1,15	0,27	22,73	25,67
3010	73702	73702	0,00	77065	75359	2,21	2,25	48,36	55,35
3011	35190	35190	0,00	38152	36054	5,50	2,45	31,09	36,78
3012	83976	83976	0,00	86008	85138	1,01	1,38	44,30	42,31
Média			0,00			1,59	1,68	40,07	39,93

Tabela I.4: Resultados detalhado GTSPR I × Gomes Jr. *et al.* (2007) (40 a 75 tarefas)

	Melhor Valor			Valor Médio				Tempo (s)	
Inst.	Gomes Jr.	GTSPR I	$\overline{imp}^{best}$ %	Gomes Jr.	GTSPR I	$\overline{imp}^{avg}$ %	$\overline{gap}^{avg}$ %	Gomes Jr.	GTSPR I
4001	57086	57086	0,00	57751	57765	-0,02	1,19	160,29	164,00
4002	49306	49306	0,00	49705	49587	0,24	0,57	108,29	144,03
4003	28643	28643	0,00	28834	28643	0,66	0,00	24,42	49,44
4004	99871	99828	0,04	102171	101837	0,33	2,01	210,85	183,61
4005	29863	29863	0,00	30594	30441	0,50	1,94	96,74	120,22
4006	32303	32303	0,00	33403	32637	2,29	1,03	83,50	91,80
4007	1E+05	116387	0,56	125443	120900	3,62	3,88	186,75	276,44
4008	44847	44847	0,00	46653	47026	-0,80	4,86	122,02	184,43
4009	77378	77378	0,00	82618	80884	2,10	4,53	189,65	212,59
4010	93591	93225	0,39	97763	98435	-0,69	5,59	230,40	245,15
4011	1E+05	105285	1,11	119856	114125	4,78	8,40	202,31	385,15
4012	1E+05	127269	0,00	135908	133552	1,73	4,94	254,26	230,48
Média			0,18			1,23	3,24	155,79	190,61
5001	105563	105426	0,13	111204	110175	0,93	4,50	585,6	683,99
5002	69631	69244	0,56	73810	72342	1,99	4,47	258,97	418,61
5003	60259	60259	0,00	61219	60548	1,10	0,48	233,27	400,68
5004	96858	96837	0,02	101444	100143	1,28	3,41	529,48	644,10
5005	80597	80140	0,57	82473	80837	1,98	0,87	323,06	440,39
5006	64500	64500	0,00	65746	65204	0,82	1,09	364,45	423,80
5007	129284	129284	0,00	140856	135675	3,68	4,94	680,62	852,58
5008	77165	76565	0,78	80279	78290	2,48	2,25	426,34	507,99
5009	55213	54234	1,77	60435	58728	2,83	8,29	584,56	690,93
5010	171912	163308	5,00	181267	177134	2,28	8,47	700,17	1004,17
5011	176332	176332	0,00	196956	187016	5,05	6,06	739,21	908,14
5012	83686	83686	0,00	93811	92443	1,46	10,46	481,63	593,82
Média			0,74			2,16	4,61	492,28	630,77
7501	258776	254755	1,55	277913	261408	5,94	2,61	1809,40	5572,48
7502	107779	106674	1,03	115331	109111	5,39	2,28	906,60	4861,01
7503	85960	85514	0,52	92820	87205	6,05	1,98	238,10	2625,59
7504	311870	300068	3,78	324516	313573	3,37	4,50	1416,50	6498,56
7505	116346	114722	1,40	130908	121305	7,34	5,74	1201,70	4297,68
7506	142313	139119	2,24	152408	142931	6,22	2,74	608,40	2784,47
7507	332419	320697	3,53	369624	330764	10,51	3,14	1876,10	12394,99
7508	218812	206014	5,85	255181	231806	9,16	12,52	1838,50	4450,51
7509	202863	175759	13,36	224472	198164	11,72	12,75	1417,60	6490,84
7510	359904	320983	10,81	392750	345894	11,93	7,76	1889,00	11881,95
7511	221263	185530	16,15	274439	215045	21,64	15,91	1673,90	7610,41
7512	137306	125184	8,83	155229	133749	13,84	6,84	1541,30	6236,35
Média			5,75			9,43	6,56	1368,09	6308,74

I.2 Execução do GTSPR II

As Tabelas I.5 a I.8 mostram os resultados detalhados para os problemas-teste de 08 a 75 tarefas, para o algoritmo GTSPR II.

As colunas representadas pelas expressões imp^{best}, imp^{avg} e gap^{avg}, são obtidas pelas Equações (5.1), (I.1) e (5.2),respectivamente.

Tabela I.5: Resultados detalhados GTSPR II × Gomes Jr. *et al.* (2007) (8 a 10 tarefas)

	Melhor Valor			Valor Médio				Tempo (s)	
Inst.	Gomes Jr.	GTSPR II	$\overline{imp}^{best}$ %	Gomes Jr.	GTSPR II	$\overline{imp}^{avg}$ %	$\overline{gap}^{avg}$ %	Gomes Jr.	GTSPR II
0801	4928	4928	0,00	4928	4928	0,00	0,00	0,05	0,03
0802	2739	2739	0,00	2739	2739	0,00	0,00	0,03	0,02
0803	4074	4074	0,00	4074	4074	0,00	0,00	0,02	0,02
0804	17149	17149	0,00	17149	17149	0,00	0,00	0,05	0,03
0805	6195	6195	0,00	6195	6195	0,00	0,00	0,03	0,02
0806	4004	4004	0,00	4004	4004	0,00	0,00	0,03	0,03
0807	10689	10689	0,00	10689	10689	0,00	0,00	0,05	0,03
0808	14059	14059	0,00	14059	14059	0,00	0,00	0,04	0,03
0809	13705	13705	0,00	13751	13705	0,33	0,00	0,05	0,03
0810	19855	19855	0,00	19855	19855	0,00	0,00	0,05	0,03
0811	22774	22774	0,00	22774	22774	0,00	0,00	0,04	0,03
0812	6184	6184	0,00	6184	6184	0,00	0,00	0,03	0,03
Média			0,00			0,03	0,00	0,04	0,03
0901	11801	11801	0,00	11801	11801	0,00	0,00	0,07	0,03
0902	8561	8561	0,00	8561	8561	0,00	0,00	0,05	0,03
0903	4734	4734	0,00	4734	4734	0,00	0,00	0,05	0,03
0904	31100	31100	0,00	31124	31100	0,08	0,00	0,08	0,07
0905	2986	2986	0,00	2986	2986	0,00	0,00	0,06	0,03
0906	13845	13845	0,00	13845	13845	0,00	0,00	0,06	0,03
0907	16400	16400	0,00	16512	16400	0,68	0,00	0,08	0,03
0908	20817	20817	0,00	20817	20817	0,00	0,00	0,07	0,04
0909	14431	14431	0,00	14431	14431	0,00	0,00	0,05	0,03
0910	25664	25664	0,00	25664	25664	0,00	0,00	0,09	0,03
0911	33202	33202	0,00	33202	33202	0,00	0,00	0,08	0,03
0912	13068	13068	0,00	13068	13068	0,00	0,00	0,06	0,03
Média			0,00			0,06	0,00	0,07	0,03
1001	14411	14411	0,00	14456	14411	0,31	0,00	0,13	0,06
1002	8349	8349	0,00	8349	8349	0,00	0,00	0,11	0,05
1003	11605	11605	0,00	11605	11605	0,00	0,00	0,10	0,05
1004	12486	12486	0,00	12486	12486	0,00	0,00	0,12	0,06
1005	5679	5679	0,00	5679	5735	-0,98	0,98	0,12	0,05
1006	2897	2897	0,00	2897	2897	0,00	0,00	0,09	0,05
1007	5515	5515	0,00	5515	5515	0,00	0,00	0,12	0,06
1008	9534	9534	0,00	9534	9534	0,00	0,00	0,10	0,04
1009	9461	9461	0,00	9461	9461	0,00	0,00	0,09	0,04
1010	22273	22273	0,00	22273	22273	0,00	0,00	0,12	0,05
1011	20514	20514	0,00	20514	20514	0,00	0,00	0,09	0,06
1012	45554	45554	0,00	45554	45554	0,00	0,00	0,10	0,06
Média			0,00			-0,06	0,08	0,11	0,05

Tabela I.6: Resultados detalhados GTSPR II × Gomes Jr. *et al.* (2007) (11 a 15 tarefas)

	Melhor Valor			Valor Médio				Tempo (s)	
Inst.	Gomes Jr.	GTSPR II	$\overline{imp}^{best}$ %	Gomes Jr.	GTSPR II	$\overline{imp}^{avg}$ %	$\overline{gap}^{avg}$ %	Gomes Jr.	GTSPR II
1101	16530	16530	0,00	16530	16530	0,00	0,00	0,16	0,07
1102	13252	13252	0,00	13252	13252	0,00	0,00	0,18	0,07
1103	15251	15251	0,00	15251	15251	0,00	0,00	0,14	0,06
1104	17573	17573	0,00	17573	17573	0,00	0,00	0,19	0,07
1105	9347	9347	0,00	9347	9451	-1,11	1,11	0,19	0,07
1106	15737	15737	0,00	15737	15737	0,00	0,00	0,22	0,08
1107	12311	12311	0,00	12311	12311	0,00	0,00	0,21	0,09
1108	17361	17361	0,00	17374	17361	0,07	0,00	0,21	0,10
1109	18202	18202	0,00	18337	18356	-0,10	0,84	0,21	0,10
1110	28886	28886	0,00	28977	28939	0,13	0,18	0,23	0,11
1111	30328	30328	0,00	30328	30328	0,00	0,00	0,20	0,10
1112	76689	76689	0,00	76911	76708	0,26	0,02	0,24	0,12
Média			0,00			-0,06	0,18	0,20	0,09
1201	8137	8137	0,00	8183	8333	-1,84	2,41	0,26	0,14
1202	9848	9848	0,00	9858	9875	-0,18	0,28	0,23	0,09
1203	8002	8002	0,00	8019	8002	0,21	0,00	0,23	0,10
1204	29466	29466	0,00	29477	29493	-0,05	0,09	0,35	0,14
1205	8468	8468	0,00	8468	8532	-0,75	0,75	0,27	0,11
1206	10982	10982	0,00	10982	10982	0,00	0,00	0,22	0,09
1207	26939	26939	0,00	27033	27093	-0,22	0,57	0,36	0,15
1208	11335	11335	0,00	11335	11335	0,00	0,00	0,27	0,14
1209	28610	28610	0,00	28967	28726	0,83	0,40	0,31	0,15
1210	23406	23406	0,00	23420	23563	-0,61	0,67	0,37	0,15
1211	23858	23858	0,00	23858	23858	0,00	0,00	0,31	0,13
1212	41983	41983	0,00	41983	41983	0,00	0,00	0,29	0,15
Média			0,00			-0,22	0,43	0,29	0,13
1501	18276	18276	0,00	18276	18276	0,00	0,00	0,64	0,23
1502	19622	19622	0,00	20198	20486	-1,42	4,40	1,23	0,34
1503	11505	11505	0,00	11952	12204	-2,11	6,07	1,09	0,32
1504	15164	15164	0,00	15250	15164	0,56	0,00	1,00	0,31
1505	12924	12924	0,00	13141	13275	-1,02	2,71	0,79	0,23
1506	9396	9396	0,00	9423	9396	0,29	0,00	0,60	0,28
1507	46544	46544	0,00	46977	48238	-2,68	3,64	1,23	0,35
1508	24899	24899	0,00	26201	26762	-2,14	7,48	1,19	0,40
1509	14457	14457	0,00	14457	14457	0,00	0,00	0,56	0,23
1510	33128	33128	0,00	33556	33343	0,63	0,65	1,11	0,42
1511	42522	42522	0,00	42866	42522	0,80	0,00	1,12	0,44
1512	12793	12793	0,00	12793	12793	0,00	0,00	0,68	0,30
Média			0,00			-0,59	2,08	0,94	0,32

Tabela I.7: Resultados detalhados GTSPR II × Gomes Jr. *et al.* (2007) (20 a 30 tarefas)

	Melhor Valor			Valor Médio				Tempo (s)	
Inst.	Gomes Jr.	GTSPR II	$\overline{imp}^{best}$ %	Gomes Jr.	GTSPR II	$\overline{imp}^{avg}$ %	$\overline{gap}^{avg}$ %	Gomes Jr.	GTSPR II
2001	16294	16294	0,00	16294	16294	0,00	0,00	1,95	0,63
2002	18107	18107	0,00	18660	18690	-0,16	3,22	3,63	1,04
2003	20249	20249	0,00	20461	20523	-0,30	1,35	3,07	0,94
2004	29941	29941	0,00	30283	30305	-0,07	1,22	4,43	1,26
2005	59158	59158	0,00	59736	59828	-0,15	1,13	4,60	1,60
2006	31053	31053	0,00	32456	32869	-1,27	5,85	4,86	1,46
2007	40438	40438	0,00	42005	41970	0,08	3,79	4,84	1,66
2008	29186	29186	0,00	29368	29329	0,13	0,49	3,18	0,92
2009	67827	67827	0,00	69246	69638	-0,57	2,67	5,61	2,01
2010	34283	34283	0,00	34656	35340	-1,97	3,08	5,52	1,52
2011	54538	54538	0,00	54911	55168	-0,47	1,15	5,90	1,35
2012	79599	79599	0,00	80171	80086	0,11	0,61	4,58	1,51
Média			0,00			-0,39	2,05	4,35	1,32
2501	23397	23397	0,00	23480	23397	0,35	0,00	13,27	4,56
2502	48540	48540	0,00	48801	49723	-1,89	2,44	13,24	3,92
2503	18503	18503	0,00	18526	18612	-0,46	0,59	9,32	2,85
2504	25645	25645	0,00	25785	25774	0,04	0,50	13,44	3,31
2505	35865	35865	0,00	36796	36294	1,37	1,20	14,14	4,69
2506	14189	14189	0,00	14219	14247	-0,19	0,41	7,85	2,34
2507	37313	37313	0,00	37857	37441	1,10	0,34	15,26	4,04
2508	44638	44638	0,00	45150	44725	0,94	0,19	13,23	4,77
2509	12839	12839	0,00	12874	12926	-0,41	0,68	10,52	3,70
2510	51415	51415	0,00	57174	56951	0,39	10,77	18,25	5,89
2511	44808	44808	0,00	47610	48314	-1,48	7,82	15,87	4,82
2512	34197	34197	0,00	35255	37232	-5,61	8,88	15,10	4,42
Média			0,00			-0,49	2,82	13,29	4,11
3001	43673	43673	0,00	44778	45494	-1,60	4,17	38,91	11,47
3002	41983	41983	0,00	42111	42356	-0,58	0,89	39,48	10,42
3003	21951	21951	0,00	22375	22640	-1,18	3,14	24,60	7,52
3004	64943	64943	0,00	68138	68012	0,18	4,73	57,49	13,09
3005	74100	74100	0,00	76040	76219	-0,24	2,86	46,12	11,04
3006	29829	29829	0,00	30832	31223	-1,27	4,67	33,51	9,13
3007	74336	74336	0,00	75378	75977	-0,79	2,21	41,04	13,57
3008	69770	69770	0,00	74097	75045	-1,28	7,56	53,20	14,99
3009	21335	21335	0,00	21641	21430	0,98	0,44	22,73	8,91
3010	73702	73702	0,00	77065	76423	0,83	3,69	48,36	15,18
3011	35190	35190	0,00	38152	36080	5,43	2,53	31,09	12,34
3012	83976	83976	0,00	86008	88115	-2,45	4,93	44,30	11,92
Média			0,00			-0,16	3,48	40,07	11,63

Tabela I.8: Resultados detalhado GTSPR II × Gomes Jr. *et al.* (2007) (40 a 75 tarefas)

	Melhor Valor			Valor Médio				Tempo (s)	
Inst.	Gomes Jr.	GTSPR II	$\overline{imp}^{best}$ %	Gomes Jr.	GTSPR II	$\overline{imp}^{avg}$ %	$\overline{gap}^{avg}$ %	Gomes Jr.	GTSPR II
4001	57086	57086	0,00	57751	58647	-1,55	2,73	160,29	41,36
4002	49306	49306	0,00	49705	49710	-0,01	0,82	108,29	31,44
4003	28643	28643	0,00	28834	28734	0,35	0,32	24,42	9,79
4004	99871	99958	-0,09	102171	103701	-1,50	3,88	210,85	51,90
4005	29863	29863	0,00	30594	31172	-1,89	4,38	96,74	27,60
4006	32303	32303	0,00	33403	33226	0,53	2,86	83,50	28,44
4007	117047	115809	1,06	125443	123347	1,67	6,51	186,75	77,35
4008	44847	45016	-0,38	46653	47087	-0,93	4,99	122,02	44,48
4009	77378	77378	0,00	82618	84155	-1,86	8,76	189,65	50,66
4010	93591	93225	0,39	97763	100000	-2,29	7,27	230,40	65,16
4011	106470	105285	1,11	119856	115764	3,41	9,95	202,31	74,57
4012	127269	127269	0,00	135908	137977	-1,52	8,41	254,26	68,97
Média			0,17			-0,47	5,07	155,79	47,64
5001	105563	105563	0,00	111204	113621	-2,17	7,77	585,6	183,06
5002	69631	69244	0,56	73810	73037	1,05	5,48	258,97	109,61
5003	60259	60259	0,00	61219	61170	0,08	1,51	233,27	87,32
5004	96858	98256	-1,44	101444	102947	-1,48	6,31	529,48	182,79
5005	80597	80140	0,57	82473	82202	0,33	2,57	323,06	109,37
5006	64500	64500	0,00	65746	65425	0,49	1,43	364,45	108,38
5007	129284	130142	-0,66	140856	139967	0,63	8,26	680,62	254,06
5008	77165	76565	0,78	80279	79509	0,96	3,84	426,34	152,00
5009	55213	54234	1,77	60435	60088	0,57	10,79	584,56	205,22
5010	171912	162543	5,45	181267	178738	1,39	9,96	700,17	365,52
5011	176332	175894	0,25	196956	196028	0,47	11,45	739,21	258,43
5012	83686	83686	0,00	93811	96938	-3,33	15,84	481,63	157,65
Média			0,61			-0,08	7,10	492,28	181,12
7501	258776	251835	2,68	277913	268341	3,44	6,55	1809,4	2055,98
7502	107779	108941	-1,08	115331	113506	1,58	6,40	906,6	798,65
7503	85960	85518	0,51	92820	89363	3,72	4,50	238,1	518,65
7504	311870	306688	1,66	324516	316658	2,42	5,53	1416,5	2090,87
7505	116346	115275	0,92	130908	124542	4,86	8,56	1201,7	1508,24
7506	142313	139592	1,91	152408	142360	6,59	2,33	608,4	689,80
7507	332419	323935	2,55	369624	340627	7,84	6,21	1876,1	2872,38
7508	218812	207038	5,38	255181	238090	6,70	15,57	1838,5	2207,77
7509	202863	172563	14,94	224472	204257	9,01	18,37	1417,6	1551,93
7510	359904	326491	9,28	392750	352438	10,26	9,80	1889	3403,45
7511	221263	200971	9,17	274439	222299	19,00	19,82	1673,9	2174,02
7512	137306	126587	7,81	155229	140441	9,53	12,19	1541,3	2074,80
Média			4,65			7,08	9,65	1368,09	1828,88

Anexo II

Publicações

Lista-se a seguir os trabalhos oriundos desta pesquisa que foram apresentados em eventos nacionais e internacionais.

Trabalhos apresentados em eventos internacionais

Título: *GRASP, Tabu Search and Path Relinking for solving total earliness/tardiness single machine scheduling problem with distinct due windows and sequence-dependent setups.*

Autores: Marcone Jamilson Freitas Souza, Puca Huachi Vaz Penna, Luiz Satoru Ochi e Frederico Augusto de Cezar Almeida Gonçalves.

Evento: *XXIX CILAMCE - Iberian Latin American Congress on Computational Methods in Engineering.*

Local: Maceió/AL - Brasil.

Período: 4 a 7 de novembro de 2008.

Trabalhos apresentados em eventos nacionais

Título: Uma heurísitca híbrida para minimizar custos com antecipação e atraso em sistemas de produção com janelas de entrega e tempos de preparação dependentes da sequência.

Autores: Marcone Jamilson Freitas Souza, Puca Huachi Vaz Penna, Luiz Satoru Ochi e Frederico Augusto de Cezar Almeida Gonçalves.

Evento: SPOLM 2008 - Simpósio de Pesquisa Operacional e Logística da Marinha.

Local: Rio de Janeiro - RJ.

Período: 5 e 6 de agosto de 2008.

Título: GRASP, VND, Busca Tabu e Reconexão por Caminhos para o Problema de Sequenciamento em uma máquina com tempos de preparação dependentes da sequência de produção, janelas de entrega distintas e penalidades por antecipação e atraso da produção.

Autores: Marcone Jamilson Freitas Souza, Puca Huachi Vaz Penna, e Frederico Augusto de Cezar Almeida Gonçalves.

Evento: XL SBPO - Simpósio Brasileiro de Pesquisa Operacional 2008.

Local: João Pessoa - PB.

Período: 2 a 5 de setembro de 2008.