Universidade Estadual de Campinas
Instituto de Matemática, Estatística e Computação Científica
Departamento de Matemática Aplicada

Uma Aplicação em Esquematização de Máquinas

Luis Franco de Campos Pinto
Mestrado em Matemática Aplicada - Campinas - SP

Orientador: Prof. Dr. Antônio Carlos Moretti

Uma Aplicação em Esquematização de Máquinas

Este exemplar corresponde à redação final da dissertação devidamente corrigida e defendida por **Luis Franco de Campos Pinto** e aprovada pela comissão julgadora.

Campinas, 10 de dezembro de 2010

Prof. Dr. **Antônio Carlos Moretti**
Orientador

Banca Examinadora:
Prof. Dr. Antônio Carlos Moretti (IMECC/UNICAMP)
Prof. Dr. Reinaldo Morábito Neto (DEP/UFSCAR)
Profa. Dra. Margarida Pinheiro Mello (IMECC/UNICAMP)

Dissertação apresentada ao Instituto de Matemática, Estatística e Computação Científica, UNICAMP, como requisito parcial para obtenção do Título de **Mestre em Matemática Aplicada**.

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP
Bibliotecária: Maria Fabiana Bezerra Müller – CRB8 / 6162

Pinto, Luis Franco de Campos
P658a Uma aplicação em esquematização de máquinas/Luis Franco de Campos Pinto-- Campinas, [S.P. : s.n.], 2011.

Orientador : Antônio Carlos Moretti
Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Computação Científica.

1.Maquinaria - Modelos matemáticos. 2.Planejamento da produção. 3.Programação inteira. 4.Programação heurística. 5.Pesquisa operacional. 6.Otimização combinatória. I. Moretti, Antônio Carlos. II. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. III. Título.

Título em inglês: An application in machine scheduling

Palavras-chave em inglês (Keywords): 1. Machinery – Mathematical models. 2. Production planning. 3. Integer programming. 4. Heuristic programming. 5. Operational research. 6. Combinatorial optimization.

Área de concentração: Pesquisa Operacional

Titulação: Mestre em Matemática Aplicada

Banca examinadora: Prof. Dr. Antônio Carlos Moretti (IMECC – UNICAMP)
Profª. Drª. Margarida Pinheiro Mello (IMECC – UNICAMP)
Prof. Dr. Reinaldo Morábito Neto (UFSCAR)

Data da defesa: 10/12/2010

Programa de Pós-Graduação: Mestrado em Matemática Aplicada

Dissertação de Mestrado defendida em 10 de dezembro de 2010 e aprovada

Pela Banca Examinadora composta pelos Profs. Drs.

Prof.(a). Dr(a). ANTONIO CARLOS MORETTI

Prof.(a). Dr(a). REINALDO MORÁBITO NETO

Prof.(a). Dr(a). MARGARIDA PINHEIRO MELLO

Agradecimentos

Agradeço:

a Deus pela minha vida,

aos meus pais Célia e Celso pelo apoio e amor incondicional,

ao meu irmão Pedro por sua amizade e incentivo,

à Patricia Conrado pelo carinho e paciência,

ao professor Moretti pela sua orientação, bom humor e amizade,

aos amigos pelo convívio ao longos destes anos,

aos meus professores que contribuíram para a realização deste trabalho,

à Emplastic e ao Maurício Borges pela oportunidade de estudar um problema estimulante,

à UniSoma por apoiar minha formação acadêmica,

ao IMECC.

Resumo

Neste trabalho, foi desenvolvida uma aplicação prática de técnicas da pesquisa operacional para a resolução de um problema real de esquematização ou programação de máquinas. Este problema deriva de um *flexible job shop scheduling*, porém apresentando diversas características próprias, impossibilitando a aplicação de modelos disponíveis na literatura. O desempenho da utilização da combinação de um modelo de programação linear inteira mista com uma heurística de construção e uma heurística de melhoramento foi avaliado diante de cenários reais obtidos da indústria de produção de frascos plásticos. Estas técnicas provaram ser eficientes para a resolução dos casos propostos.

Palavras-chave: Maquinaria - Modelos matemáticos, Planejamento da Produção, Programação inteira, Programação heurística, Pesquisa Operacional, Otimização combinatória.

Abstract

In this work, a practical application of operational research techniques was developed to solve a real machine scheduling or programming problem. This problem derives from a flexible job shop scheduling framework, but presents several unique characteristics, which makes it impossible to apply models available in literature. The performance of using a combination of a mixed integer programming model with a construction heuristic and a improvement heuristic was evaluated using real world scenarios obtained from the plastic bottle production industry. Theses techniques were proven efficient in resolving the proposed cases.

Keywords: Machinery - Mathematical models, Production planning, Integer programming, Heuristic programming, Operational research, Combinatorial optimization.

Conteúdo

Agradecimentos **v**

Resumo **vii**

1 Introdução **1**

2 O problema de programação de máquinas e suas variações **5**
2.1 Introdução 5
2.1.1 Exemplo: Fábrica de sacos de papel 5
2.1.2 Exemplo: Escolha de portões num aeroporto 6
2.1.3 Exemplo: Programação de tarefas num CPU 7
2.2 *Framework* e notação 7
2.2.1 Ambiente de máquinas (α) 8
2.2.1.1 Uma máquina (1) 8
2.2.1.2 Máquinas idênticas em paralelo (Pm) 8
2.2.1.3 Máquinas com diferentes velocidades em paralelo (Qm) 8
2.2.1.4 Máquinas não relacionadas em paralelo (Rm) . . 8
2.2.1.5 *Flow shop* (Fm) 8
2.2.1.6 *Flexible flow shop* (FFs) 9
2.2.1.7 *Open shop* (Om) 9
2.2.1.8 *Job Shop* (Jm) 9
2.2.1.9 *Flexible Job Shop* (FJs) 9
2.2.2 Características de produção (β) 10
2.2.2.1 Data de início (r_j) 10
2.2.2.2 Tempo de *setup* entre tarefas (s_{j_1,j_2}) 10
2.2.2.3 Preemptividade ($prmp$) 10
2.2.2.4 Restrições de precedência ($prec$) 10

2.2.2.5 Quebra de máquinas (*brkdwn*) 11
2.2.2.6 Restrições de máquina-tarefa (M_j) 11
2.2.2.7 Permutação (*prmu*) 11
2.2.2.8 Bloqueio (*block*) 11
2.2.2.9 Sem espera (*nwt*) 12
2.2.2.10 Recirculação (*recrc*) 12
2.2.2.11 *Tooling* (*tool*) 12
2.2.3 Objetivo (γ) 12
2.2.3.1 *Makespan* (C_{max}) 12
2.2.3.2 Menor atraso (L_{max}) 12
2.2.3.3 Tempo total de término ponderado ($\sum w_j C_j$) . . 13
2.2.3.4 Tempo total descontado de término ponderado ($\sum w_j(1 - e^{-rC_j})$) 13
2.2.3.5 Atraso total ponderado ($\sum w_j L_j$) 13
2.2.3.6 Total de tarefas atrasadas ponderada ($\sum w_j U_j$) . 13

3 Definição do problema 15
3.1 Introdução 15
3.2 Condições do problema 20
3.2.1 Data de início de produção 20
3.2.2 Data de entrega do produto 20
3.2.3 Demanda por produtos de setores variados 20
3.2.4 Estoque inicial 20
3.2.5 Produção ininterrupta 20
3.2.6 Produção multi-máquinas 21
3.2.7 Tempo de *setup* 21
3.2.8 Recurso limitado: funcionários 21
3.2.9 Recurso limitado: materiais 21
3.2.10 Etapas de produção 21
3.2.11 Turnos de produção variados 22
3.2.12 Possibilidade de hora extra 22
3.2.13 Tarefas predecessoras compartilhadas 22
3.3 Classificando o problema 22
3.3.1 Ambiente de máquinas 23
3.3.2 Características de Produção 23
3.3.3 Objetivo 23
3.4 Resoluções propostas 24

4 Modelo Matemático **27**
4.1 Introdução . . . 27
4.2 Parâmetros do Modelo . . . 27
4.3 Variáveis . . . 32
4.4 Restrições . . . 32
4.4.1 Definição do período de produção . . . 33
4.4.2 Atendimento da demanda . . . 34
4.4.3 Restrições operacionais de produção . . . 34
4.4.4 Produção por ligação de precedência . . . 36
4.4.5 Definição do término da tarefa . . . 36
4.5 Função objetivo . . . 36

5 Heurísticas Aplicadas **37**
5.1 Introdução . . . 37
5.2 Heurística de construção . . . 38
5.3 Aplicação: solução inicial . . . 39
5.3.1 Alocação de tarefas com demanda direta . . . 41
5.3.2 Alocação de tarefas com demanda indireta (com predecessores) . . . 43
5.3.3 Alocação de tarefas com demanda indireta (sem predecessores) . . . 48
5.3.4 Exemplo . . . 51
5.4 Heurística de melhoramento . . . 52
5.5 Aplicação: melhoramento de uma solução . . . 53
5.5.1 Melhoria do término total das tarefas da solução . . . 53
5.5.2 Melhoria na factibilidade . . . 57
5.5.3 Exemplo . . . 59

6 Resultados Computacionais **61**
6.1 Introdução . . . 61
6.2 Cenários . . . 61
6.3 Resultados . . . 62
6.3.1 Emplastic_0809 . . . 65
6.3.2 Emplastic_1209 . . . 70
6.3.3 Emplastic_0210 . . . 72
6.4 Análise . . . 79

7 Conclusões e trabalhos futuros **81**

Bibliografia **83**

A Cenários **85**
A.1 Emplastic_0809 85
A.2 Emplastic_1209 95
A.3 Emplastic_0210 107

Lista de Figuras

1.1 Exemplo de diagrama de Gantt 2

3.1 Frascos produzidos pela Emplastic 16
3.2 Molde utilizado no processo de sopro 17
3.3 Processo de sopro 18
3.4 Processo de serigrafia 19
3.5 Representação do problema de *job shop* na forma de grafo 24

6.1 Emplastic_0809: Evolução da melhor solução inteira na configuração *Melhoramento* 66
6.2 Emplastic_0809: Evolução da melhor solução inteira na configuração *Construção* 68
6.3 Emplastic_0809: Evolução da melhor solução inteira na configuração *Modelo* 70
6.4 Emplastic_1209: Evolução da melhor solução inteira na configuração *Construção* 72
6.5 Emplastic_0210: Evolução da melhor solução inteira na configuração *Melhoramento* 75
6.6 Emplastic_0210: Evolução da melhor solução inteira na configuração *Construção* 77
6.7 Emplastic_0210: Evolução da melhor solução inteira na configuração *Modelo* 79

Lista de Tabelas

4.1 Dias dos turnos do *sopro* 31
4.2 Dias dos turnos do *serigrafia* 31
4.3 Relação entre turnos de *sopro* e *serigrafia* 31

5.1 Exemplo para heurística de construção: primeira etapa 52
5.2 Exemplo para heurística de construção: segunda etapa 52
5.3 Exemplo para heurística de construção: terceira etapa 52
5.4 Exemplo para heurística de melhoramento: primeira etapa 59
5.5 Exemplo para heurística de melhoramento: primeira etapa 60

6.1 Emplastic_0809: Penalização de entrega ($\alpha_{3,j}$) 62
6.2 Emplastic_1209: Penalização de entrega ($\alpha_{3,j}$) 63
6.3 Emplastic_0210: Penalização de entrega ($\alpha_{3,j}$) 64
6.4 Emplastic_0809: Resultado obtido 65
6.5 Emplastic_0809: Evolução das melhores soluções factíveis para configuração *Melhoramento* 66
6.6 Emplastic_0809: Evolução das melhores soluções factíveis para configuração *Construção* 67
6.7 Emplastic_0809: Evolução das melhores soluções factíveis para configuração *Modelo* 69
6.8 Emplastic_1209: Resultado obtido 71
6.9 Emplastic_1209: Evolução das melhores soluções factíveis para configuração *Construção* 71
6.10 Emplastic_0210: Resultado obtido 73
6.11 Emplastic_0210: Evolução das melhores soluções factíveis para configuração *Melhoramento* 74
6.12 Emplastic_0210: Evolução das melhores soluções factíveis para configuração *Construção* 76

6.13 Emplastic_0210: Evolução das melhores soluções factíveis para configuração *Modelo* 78

A.1 Emplastic_0809: Produtividade ($Prod_{j,m}$) [unidades/h] 86
A.2 Emplastic_0809: Tempo de *setup* de tarefas do setor de serigrafia ($Setup_{j_i,j_k}$) [h] 88
A.3 Emplastic_0809: Tempo de *setup* de tarefas do setor de sopro ($Setup_{j_i,j_k}$) [h] 89
A.4 Emplastic_0809: Demanda (Dem_j) [unidades] 89
A.5 Emplastic_0809: Ligação de tarefas ($LigTarefa_{j_i,j_k}$) 90
A.6 Emplastic_0809: Data entrega (DE_j) 90
A.7 Emplastic_0809: Funcionários para qualquer tarefa nas máquinas ($F_{j,m}$) 91
A.8 Emplastic_0809: Moldes disponíveis (NM_j) 92
A.9 Emplastic_0809: Tarefa usa material ($JMat_{j,r}$) 93
A.10 Emplastic_0809: Setor de tarefa ($JSetor_{j,s}$) 93
A.11 Emplastic_0809: Horas normais e extras disponíveis por semana na serigrafia ($HTurno_{t,s}$ e $HExt_{t,s}$) [h] 95
A.12 Emplastic_0809: Horas normais e extras disponíveis por semana no setor sopro ($HTurno_{t,s}$ e $HExt_{t,s}$) [h] 95
A.13 Emplastic_0809: Estoque inicial (Est_j) [unidades] 95
A.14 Emplastic_1209: Produtividade ($Prod_{j,m}$) [unidades/h] 96
A.15 Emplastic_1209: Tempo de *setup* de tarefas do setor de serigrafia ($Setup_{j_i,j_k}$) [h] 98
A.16 Emplastic_1209: Tempo de *setup* de tarefas do setor de sopro ($Setup_{j_i,j_k}$) [h] 99
A.17 Emplastic_1209: Demanda (Dem_j) [unidades] 100
A.18 Emplastic_1209: Ligação de tarefas ($LigTarefa_{j_i,j_k}$) 101
A.19 Emplastic_1209: Data entrega (DE_j) 101
A.20 Emplastic_1209: Funcionários para qualquer tarefa nas máquinas ($F_{j,m}$) 102
A.21 Emplastic_1209: Moldes disponíveis (NM_j) 103
A.22 Emplastic_1209: Tarefa usa material ($JMat_{j,r}$) 104
A.23 Emplastic_1209: Setor de tarefa ($JSetor_{j,s}$) 105
A.24 Emplastic_1209: Horas normais e extras disponíveis por semana na serigrafia ($HTurno_{t,s}$ e $HExt_{t,s}$) [h] 107
A.25 Emplastic_1209: Horas normais e extras disponíveis por semana no setor sopro ($HTurno_{t,s}$ e $HExt_{t,s}$) [h] 107

A.26 Emplastic_0210: Produtividade ($Prod_{j,m}$) [unidades/h] 108
A.27 Emplastic_0210: Tempo de *setup* de tarefas do setor de serigrafia ($Setup_{j_i,j_k}$) [unidades/h] 110
A.28 Emplastic_0210: Tempo de *setup* de tarefas do setor de sopro ($Setup_{j_i,j_k}$) [h] 110
A.29 Emplastic_0210: Demanda (Dem_j) [unidades] 111
A.30 Emplastic_0210: Ligação de tarefas ($LigTarefa_{j_i,j_k}$) 112
A.31 Emplastic_0210: Data entrega (DE_j) 112
A.32 Emplastic_0210: Funcionários para qualquer tarefa nas máquinas de Sopro ($F_{j,m}$) 113
A.33 Emplastic_0210: Funcionários para as tarefas nas máquinas de Serigrafia ($F_{j,m}$) 113
A.34 Emplastic_0210: Moldes disponíveis (NM_j) 114
A.35 Emplastic_0210: Tarefa usa material ($JMat_{j,r}$) 115
A.36 Emplastic_0210: Setor de tarefa ($JSetor_{j,s}$) 115
A.37 Emplastic_1209: Horas normais e extras disponíveis por semana na serigrafia ($HTurno_{t,s}$ e $HExt_{t,s}$) 116
A.38 Emplastic_1209: Horas normais e extras disponíveis por semana no setor sopro ($HTurno_{t,s}$ e $HExt_{t,s}$) 117

Capítulo 1

Introdução

O problema de esquematização de máquinas[1] tem sido estudado exaustivamente desde o início das fábricas na revolução industrial. Um dos pioneiros no estudo de *scheduling* foi Henry Gantt, criador do "diagrama de Gantt" (exemplificada na figura 1.1, que mostra uma programação da produção), tabela que auxilia na geração e visualização de planos de produção. Em seu livro "Work, Wages and Profit" (Gantt [1913]), ele relata a necessidade da programação sequencial de tarefas para serem realizadas a cada dia, o que deveria aumentar a eficiência da produção.

A programação para uma linha de produção com recursos limitados pode ter impactos graves ao ponto de poder determinar a sobrevivência ou não de uma empresa de menor porte, especialmente em indústrias altamente competitivas e com forte concorrência. O objetivo dessa programação é definir a programação da produção de maneira a melhorar a eficiência global, garantindo prazos de entrega e minimizando os custos totais.

Apesar do interesse antigo, publicações propondo modelos para a resolução deste problema começaram a aparecer apenas no fim da década de 1950 e começo da década de 1960 (veja Jackson [1956], Wagner [1959] e Manne [1960], como exemplo). Ainda que muito estudado, o problema continua sendo considerado de complexidade elevada, dada a origem combinatorial do problema.

Job shop scheduling, uma das mais usuais configurações de esquematização de máquinas, consiste no agendamento de tarefas de tamanhos diferentes em máquinas idênticas. Este problema, com objetivo de minimizar o término da

[1] Também conhecido como "problema de programação da produção" ou, em inglês, *scheduling*.

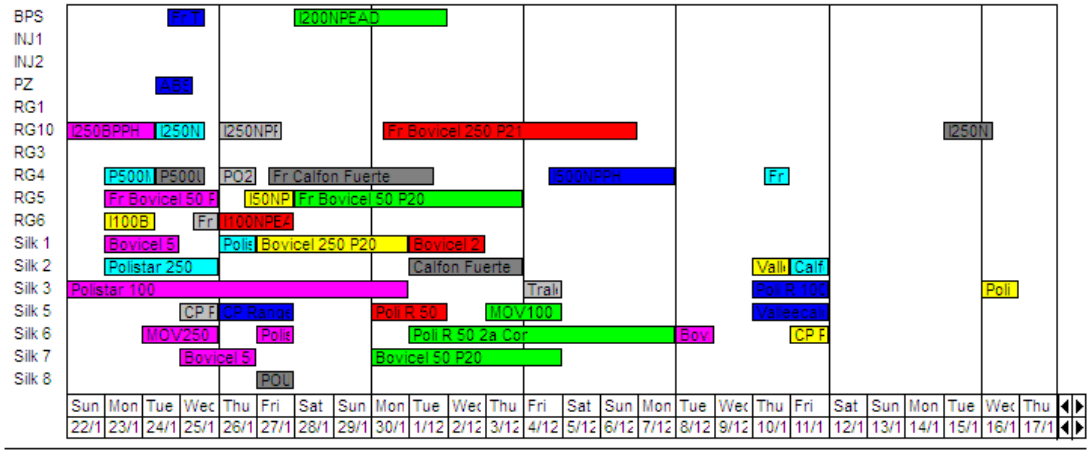

Figura 1.1: Exemplo de diagrama de Gantt

última tarefa, é NP-completo[2], como apresentado em Garey et al. [1976].

Além desta configuração do problema, existem muitas outras variações, que foram geradas visando uma maior aderência aos planejamentos realizados em fábricas reais. Para citar alguns, temos: *flow shop scheduling*, *flexible job shop scheduling*, *open shop scheduling*, restrições de ferramentas, tempos (custos) de *setup* etc.

Para a resolução destes problemas, muitas abordagens já foram exploradas, como modelos matemáticos (programação linear, inteira e não-linear), meta-heurísticas (busca tabu, algoritmos genéticos, s*imulated annealing* etc.), heurísticas de construção, heurísticas de melhoramento, etc.

Esta dissertação tem como objetivo apresentar uma aplicação de técnicas de pesquisa operacional para um caso prático de programação da produção, desenvolvido junto à empresa *Emplastic Ind. Com. de Plásticos Ltda.* para o agendamento de máquinas e tarefas na produção de embalagens plásticas. Um modelo matemático de programação linear inteira mista foi desenvolvido. Heurísticas complementares foram abordadas como auxílio. Estes algoritmos foram avaliados com relação a sua performance (de acordo com uma função objetiva, apresentada na dissertação) a partir de casos reais e mostrou-se que é possível obter soluções factíveis para os cenários. No entanto, não foi possível concluir

[2] Na teoria da complexidade, problemas NP-completo tem alto grau de complexidade e não foram encontradas provas que os mesmos podem ser resolvidos de maneira eficiente (complexidade polinomial).

qual abordagem de otimização apresenta melhor desempenho prático devido ao baixo número de casos para teste.

A dissertação é apresentada da seguinte maneira: no capítulo 2, o problema de programação de máquinas é apresentado, assim como uma série de variações possíveis para o problema, além de definir um método de classificação do mesmo; no capítulo 3, caracterizamos o problema tratado por esta dissertação, definindo suas condições e classificando-o; no capítulo 4, apresentamos o modelo matemático desenvolvido para o problema proposto; no capítulo 5, apresentamos as duas heurísticas desenvolvidas para auxiliar na resolução do problema, uma de construção e a outra de melhoramento; no capítulo 6, descrevemos os cenários reais utilizados para testes da aplicação e os resultados obtidos; no capítulo 7 são apresentadas as conclusões e propostas para trabalhos futuros.

Capítulo 2

O problema de programação de máquinas e suas variações

2.1 Introdução

O problema de esquematização de máquinas pode ser generalizado para a alocação de recursos a tarefas ao longo de um horizonte de planejamento. Esta alocação visa otimizar um ou mais critérios de performance, obedecendo a uma série de restrições operacionais ou estratégicas do negócio.

Os recursos que compõem o problema podem ter origens bastante variadas como máquinas numa fábrica, pistas e terminais num aeroporto, processadores num computador, etc. e suas tarefas associadas, como a produção de um determinado componente na fábrica, aterrissagens e decolagem num aeroporto (e seus respectivos embarques e desembarques), a ordem de execução de programas num computador, etc.

Para exemplificar, vamos apresentar três exemplos encontrados em Pinedo [1995].

2.1.1 Exemplo: Fábrica de sacos de papel

Considere uma fábrica que produz sacos de papel para cimento, carvão, ração animal, etc. O principal insumo para a fábrica são os rolos de papel. O processo de produção consiste em três etapas: imprimir o logo no papel, colar os lados do saco e costurar uma ou ambas as pontas do saco. Em cada etapa de produção temos diversas máquinas que podem ser diferentes em termos de velocidades, número de cores de impressão ou tamanho de sacos que podem utilizar. Cada pedido para

a fábrica inclui o número de sacos que devem ser entregues juntamente com uma data de entrega. Um atraso na entrega implica a penalidade, na forma de perda de confiança do cliente, que pode ser maior ou menor dependendo da importância do cliente e/ou produto entregue, além do tempo de atraso. Um dos objetivos no planejamento de produção é minimizar estas penalidades.

Outras restrições vem do processo de produção, como um tempo de *setup*[1] quando há troca de tipos de sacos sendo processados em uma máquina. O tempo de *setup* depende da ordem da mudança dos sacos. Outro objetivo do planejamento é minimizar o tempo gasto com *setups*.

2.1.2 Exemplo: Escolha de portões num aeroporto

Considere um grande terminal de um aeroporto. Existem dezenas de portões de decolagem e aterrizagem, além de centenas de aviões que pousam e decolam todos os dias. Os portões e os aviões não são idênticos. Alguns portões estão localizados de maneira a atender a aviões de grande porte, enquanto que outros portões são de difícil acesso, ao ponto do avião precisar ser rebocado até o portão.

Aviões embarcam e desembarcam de acordo com uma programação. Porém, esta programação está sujeita a variações aleatórias, como mudanças climáticas, problemas em outros aeroportos, nos próprios aviões, etc. Quando um avião é levado a um portão, os passageiros devem ser desembarcados, o avião deve passar por serviços de limpeza e reabastecido e, por fim, os (novos) passageiros devem ser embarcados. A hora de partida do vôo pode ser encarada como uma data limite e atrasos são encarados como performance negativa. No entanto, devido à política de conservação de combustível, um avião não pode decolar se não puder pousar no seu destino como consequência de congestionamentos em outros aeroportos ou outros problemas. Em geral, passageiros aguardam no terminal do aeroporto para a liberação do avião, ao invés de embarcados no avião, impedindo assim que outros aviões utilizem o portão.

Logo, a decisão de determinar qual avião deve usar qual portão deve gerar uma programação fisicamente viável (com os portões livres para uso no momento de chegada do avião) e de maneira a otimizar alguns critérios de performance, como minimizar o número de atrasos e o menor deslocamento de funcionários do aeroporto para realizar serviços nos aviões.

Desta maneira, temos que os portões são os recursos e os aviões são as tarefas, com datas de início e fim.

[1] Tempo de *setup* refere-se ao tempo gasto na preparação da máquina que será utilizada.

2.1.3 Exemplo: Programação de tarefas num CPU

Uma das funcionalidades de um sistema operacional multi-tarefa é determinar o tempo que o CPU dedicará a cada tarefa que precisa ser realizada. Em geral, o tempo necessário para a realização de uma tarefa não é conhecida *a priori*, mas a distribuição do tempo de processamento, junto com valor esperado e variância, sim. Além disso, cada tarefa tem um fator de prioridade para término (que pode ser determinado pelo usuário). Assim, um objetivo do plano pode ser minimizar a soma do tempo de término das tarefas ponderado pela sua prioridade.

Para evitar que tarefas pequenas de baixa prioridade demorem demais para serem realizadas, o sistema operacional corta tarefas longas em partes menores. Assim o CPU dedica tempo a todas as tarefas (mais a umas que outras). Caso a tarefa seja pequena o suficiente, será realizada logo.

A interrupção de uma tarefa parcialmente realizada, sem perda do trabalho já realizado, é chamada de preemptividade.

2.2 *Framework* e notação

Na literatura, são propostas diversas variações do problema de programação de máquinas, cada uma considerando um conjunto específico de características do processo de produção, mas mantendo em comum o agendamento de um número finito de tarefas a máquinas (também finitas).

Para apresentar algumas as variações do problema, serão adotadas as notações e *frameworks* de Pinedo [1995], com a inclusão de variações adicionais, devidamente identificadas como tal, para auxiliar na classificação posterior do problema abordado nesta dissertação.

Pela classificação, um problema pode ser descrito pela tripla $\alpha \mid \beta \mid \gamma$. O campo α corresponde ao ambiente das máquinas e pode conter apenas um item. O campo β refere às características de produção e suas possíveis restrições, podendo ter nenhum, um ou mais itens. O campo γ determina o objetivo a ser minimizado pelo problema e, em geral, deve conter apenas um item.

Nas caracterizações a seguir, considere n tarefas e m máquinas. Também considere que o índice j refere-se às tarefas e i às máquinas.

2.2.1 Ambiente de máquinas (α)

2.2.1.1 Uma máquina (*1*)

Caso mais simples possível, em que existe apenas uma máquina disponível. Todos outros ambientes de máquinas podem ser particularizados para este caso especial.

2.2.1.2 Máquinas idênticas em paralelo (*Pm*)

Caso em que existem m máquinas idênticas trabalhando em paralelo. O caso geral, em que qualquer máquina pode realizar uma tarefa, é definido como Pm. No caso particular em que uma tarefa j pode ser processado apenas em um subconjunto de máquinas, digamos M_j, então usamos M_j para caracterizar o problema, no campo β.

2.2.1.3 Máquinas com diferentes velocidades em paralelo (*Qm*)

Ambiente com m máquinas de velocidades diferentes. A velocidade da máquina i é definida como v_i e o tempo de processamento da tarefa j na máquina i é dado por $p_{i,j} = \frac{p_j}{v_i}$, onde p_j é o tempo de processamento da tarefa j.

2.2.1.4 Máquinas não relacionadas em paralelo (*Rm*)

Generalização do caso anterior, com m máquinas em paralelo e velocidade de processamento para cada tarefa dado por $v_{i,j}$. Desta maneira, o tempo de processamento de cada tarefa em cada máquina é dado por $p_{i,j} = \frac{p_j}{v_{i,j}}$.

2.2.1.5 *Flow shop* (*Fm*)

Existem m máquinas em série. Cada tarefa deve ser processada em cada uma das m máquinas. Todas as tarefas tem o mesmo percurso de produção, ou seja, são processadas primeiramente na máquina 1, depois na máquina 2 e assim por diante. Quando uma tarefa é realizada em uma máquina, ela passa a compor a fila para ser processada na próxima máquina. Caso esta fila seja determinada por uma política do tipo *FIFO* (*first in, first out*)[2] o problema é chamado de *Flow Shop de Permutação* e usamos o código *prmu* para caracterizar o mesmo, no campo β.

[2] Políticas do tipo *FIFO* determinam que o elemento mais antigo da fila é o primeiro a ser retirado da mesma.

2.2.1.6 *Flexible flow shop* (*FFs*)

Este tipo de produção é uma combinação dos ambientes de *flow shop* e de máquinas em paralelo. Ao invés de passar por m máquinas em série, cada tarefa deve passar por s etapas de produção em série (a tarefa deve passar pela primeira etapa primeiro, pela segunda etapa em seguida e assim por diante). Em cada etapa existe um conjunto de máquinas disponíveis; cada tarefa j que passar por uma etapa requer apenas uma máquina entre as disponíveis e, em geral, qualquer uma destas máquinas pode realizar esta tarefa. Podem existir filas entre etapas, conforme as tarefas são liberadas nas máquinas. Estas filas geralmente são do tipo *FIFO*.

2.2.1.7 *Open shop* (*Om*)

Cada uma das n tarefas deve ser processada nas m máquinas. Porém, alguns desses tempos de processamento podem ser zero. Além disso, não existe restrição com relação à ordem em que as tarefas são processadas nas máquinas, cada uma podendo ter sua própria rota na fábrica.

2.2.1.8 *Job Shop* (*Jm*)

No caso de um *job shop* com m máquinas, cada tarefa j possui sua própria sequência (rota) de máquinas nas quais ela deve ser processada. Para o caso em que a rota de uma tarefa incluir mais de uma passagem em uma máquina, incluímos o item *recrc* no campo β, caracterizando a recirculação da tarefa na máquina.

2.2.1.9 *Flexible Job Shop* (*FJs*)

Apesar de não apresentado na classificação de Pinedo [1995], o *flexible job shop* também é um ambiente de máquinas frequente na literatura. Em Gao et al. [2007], temos que este caso é uma combinação de *job shop* (Jm) com máquinas em paralelo. Semelhante ao FFs (*flexible flow shop*), as tarefas deverão passar por um subconjunto dos s setores de produção existentes, em ordem pré-definida para cada tarefa. Em cada setor, existe um conjunto de máquinas em paralelo disponíveis, podendo ser idênticas (Pm), com velocidades diferentes (Qm) ou não relacionadas (Rm); cada tarefa deverá passar por uma das máquinas dos setores que foram previamente definidos para ela. Podem haver filas entre setores.

2.2.2 Características de produção (β)

2.2.2.1 Data de início (r_j)

Caso exista um impedimento para o início imediato de uma tarefa j, o item r_j é apresentado no campo de restrições de produção. Neste caso, cada tarefa j só poderá iniciar sua produção após r_j.

2.2.2.2 Tempo de *setup* entre tarefas (s_{j_1,j_2})

O parâmetro s_{j_1,j_2} define que existe um tempo de *setup* (preparação) entre a realização da tarefa predecessora j_1 e a tarefa sucessora j_2, caso sejam realizadas em sequência. Caso haja dependência do tempo de *setup* com relação à máquina, então um índice i é incluído no parâmetro, isto é, $s_{j_1,j_2,i}$.

2.2.2.3 Preemptividade (*prmp*)

Preemptividade define que uma tarefa j que está sendo realizada na máquina i pode ser interrompida para a realização de outra tarefa, sem a perda do trabalho que já foi realizado na tarefa original. Neste caso, quando a máquina retomar a realização da tarefa j (ou em outra máquina no caso de máquinas em paralelo) será necessário apenas concluir o trabalho faltante.

2.2.2.4 Restrições de precedência (*prec*)

Restrições de precedência impõem que uma tarefa j_1 só pode ser iniciada em uma máquina qualquer após a conclusão de uma outra tarefa j_2. Existem alguns casos particulares de restrições de precedência. No caso em que cada tarefa possui não mais que um predecessor, a restrição é chamada de *intree*[3]. No caso em que cada tarefa possui não mais que um sucessor, a restrição é chamada de *outtree*[4]. No caso em que ambas as condições são atendidas, ou seja, cada tarefa não possui mais do que um sucessor e não mais que um predecessor, a cadeia de produção é linear.

[3] Neste caso, a cadeia de produção forma uma árvore com característica de se alargar conforme se vai aprofundando.

[4] Neste caso, a cadeia de produção forma uma árvore com característica de se estreitar conforme se vai aprofundando.

2.2.2.5 Quebra de máquinas (*brkdwn*[5])

No caso em que máquinas vão ficar indisponíveis em algum momento ao longo do horizonte de planejamento, usamos o item *brkdwn* para caracterizar o problema. Assume-se, para modelos determinísticos, que o momento de indisponibilidade é conhecido a priori e pode, inclusive, ser encarado como uma manutenção preventiva programada, ou outro tipo de parada já conhecida.

2.2.2.6 Restrições de máquina-tarefa (M_j)

Esta restrição só pode existir em ambientes com máquinas paralelas, conforme já foi apresentado na seção 2.2.1.2. Quando ocorre, apenas algumas máquinas podem realizar algumas tarefas, sendo que a tarefa j pode ser realizada no subconjunto de máquinas dado por M_j.

2.2.2.7 Permutação (*prmu*)

Conforme previamente apresentado na seção 2.2.1.5 e aplicável apenas a ambientes de *flow shop*, problemas de permutação são tais que as tarefas devem respeitar filas do tipo *FIFO (first in, first out)* enquanto aguardam a liberação de uma máquina, o que implica que a ordem em que as tarefas são realizadas nas máquinas é sempre a mesma. Sendo assim o problema consiste em definir a sequência geral de realização das tarefas.

2.2.2.8 Bloqueio (*block*[6])

O bloqueio de máquinas só pode ocorrer em ambientes de *flow shop*. Esta restrição define que existe um *buffer*[7] de tamanho finito entre duas máquinas em sequência. Quando o *buffer* estiver completo entre duas máquinas (digamos i_1 e i_2), uma tarefa j sendo realizada na máquina i_1 predecessora não pode ser liberada enquanto a máquina i_2 sucessora não terminar sua tarefa. Desta maneira, a máquina i_1 fica temporariamente impedida de realizar outra tarefa. Apenas quando a máquina i_2 for liberada para realizar tarefas do *buffer* (ou a própria tarefa j que está aguardando na máquina i_1), a máquina i_2 poderá realizar uma nova tarefa.

---

5 O termo *brkdwn* vem da inglês *breakdown*, ou quebra em português.

6 Do inglês *blocking*, ou bloqueando em português.

7 O *buffer* existe quando uma tarefa tem que esperar a liberação da próxima máquina para que possa ser realizado. Pode ser na forma de pilha, fila etc.

2.2.2.9 Sem espera (*nwt*[8])

Esta característica de problemas também só poderá ocorrer em ambientes de *flow shop*. Neste caso, tarefas não podem esperar a liberação de uma máquina para seguir adiante na sequência de máquinas. Logo, o início da tarefa deve ser postergado até que seja garantido que a tarefa não irá parar entre duas máquinas. Um exemplo de ambiente que tem esta característica é no laminador de tiras a quente em siderurgias, aonde uma placa não pode esperar e esfriar entre duas máquinas.

2.2.2.10 Recirculação (*recrc*)

Conforme já descrito na seção 2.2.1.8, em problemas com recirculação, uma tarefa pode passar mais de uma vez em uma máquina.

2.2.2.11 *Tooling*[9] (*tool*)

Restrições de ferramental, mesmo não constando nas opções de Pinedo [1995], são bastante comuns na literatura. As apresentadas em Widmer [1991] são definidas quando um recurso escasso e compartilhado por máquinas deve ser levado em consideração para a elaboração de um plano de produção. Um exemplo de recurso escasso é um número limitado de operadores de máquinas, necessários para o funcionamento das mesmas, que não são suficientes para que todas as máquinas operem em conjunto num mesmo turno.

2.2.3 Objetivo (γ)

2.2.3.1 *Makespan* (C_{max})

O objetivo deste tipo de problema é minimizar o término da última tarefa a ser realizada no sistema. Se C_j é o tempo de término da tarefa j então o objetivo pode ser expresso como $Minimizar \max_j\{C_j\}$.

2.2.3.2 Menor atraso (L_{max})

O objetivo é minimizar o maior atraso do término de tarefas. Considerando que L_j é o tempo de atraso do término de uma tarefa j, então o objetivo pode ser

[8] Do inglês *no-wait*, ou sem espera em português.

[9] A palavra *tool*, em inglês, quer dizer ferramenta.

expresso como $Minimizar \max_j\{L_j\}$.

2.2.3.3 Tempo total de término ponderado ($\sum w_jC_j$)

Considerando um peso w_j para cada tarefa j e o tempo de término de cada tarefa como C_j, o objetivo deste problema é minimizar o tempo de término de realização de todas as tarefas ponderadas. Muitas vezes este objetivo é definido quando a demora para realização de uma tarefa implica num custo diferente para cada tarefa (por isto a ponderação).

2.2.3.4 Tempo total descontado de término ponderado ($\sum w_j(1-e^{-rC_j})$)

Fórmula mais generalizada de ponderação do tempo de término de cada tarefa. Implica que, se uma tarefa não estiver terminada em t, um custo adicional de $w_jre^{-rt}dt$ incorrerá durante o período $[t, t+dt]$. Em geral, $r \leq 10\%$.

2.2.3.5 Atraso total ponderado ($\sum w_jL_j$)

Soma dos atrasos (conforme definido na seção 2.2.3.2) ponderado.

2.2.3.6 Total de tarefas atrasadas ponderada ($\sum w_jU_j$)

Se definirmos $U_j = 1$ se a tarefa j está atrasada, e 0 caso contrário, definimos o objetivo deste problema como a minimização do total de atrasos ponderados (um atraso pode ser mais ou menos importante que outro).

Capítulo 3

Definição do problema

3.1 Introdução

O problema de esquematização de máquinas da empresa *Emplastic*, estudado neste trabalho, consiste em definir a programação da produção (por turnos) das máquinas disponíveis durante um horizonte de planejamento, tipicamente de 1 mês.

A fábrica em questão é especializada na fabricação de frascos e tampas plásticas. A fábrica também possui uma seção para a impressão de informações diretamente nos frascos. Os principais clientes da empresa são dos setores farmacêuticos, veterinários e cosméticos (vide Web [2010] para maiores informações).

Para a produção de frascos, existem dois processos diferentes: por injeção ou por sopro.

No processo de modelagem por injeção, os pequenos grânulos de plásticos são derretidos (ou plastificados) em um cilindro interno da máquina injetora através de resistências. Para a formação do frasco ou tampa, um bico de injeção preenche o molde, aplicando pressão para que a forma não seja distorcida. Após o resfriamento, o produto é liberado do molde e está pronto para utilização.

Já no processo de modelagem por sopro, o plastico derretido é fundido na forma de cilindro que é encaixado em um molde. Em seguida, ar é injetado em um dos lados do cilindro, num processo para inflar o cilindro e preencher completamente o molde. O tubo é resfriado e o frasco retirado do molde.

É possível encontrar um detalhamento mais preciso do processo em Rosato et al. [2000].

O processo de impressão nos frascos se dá por serigrafia ou silk-screen. Este

Figura 3.1: Frascos produzidos pela Emplastic

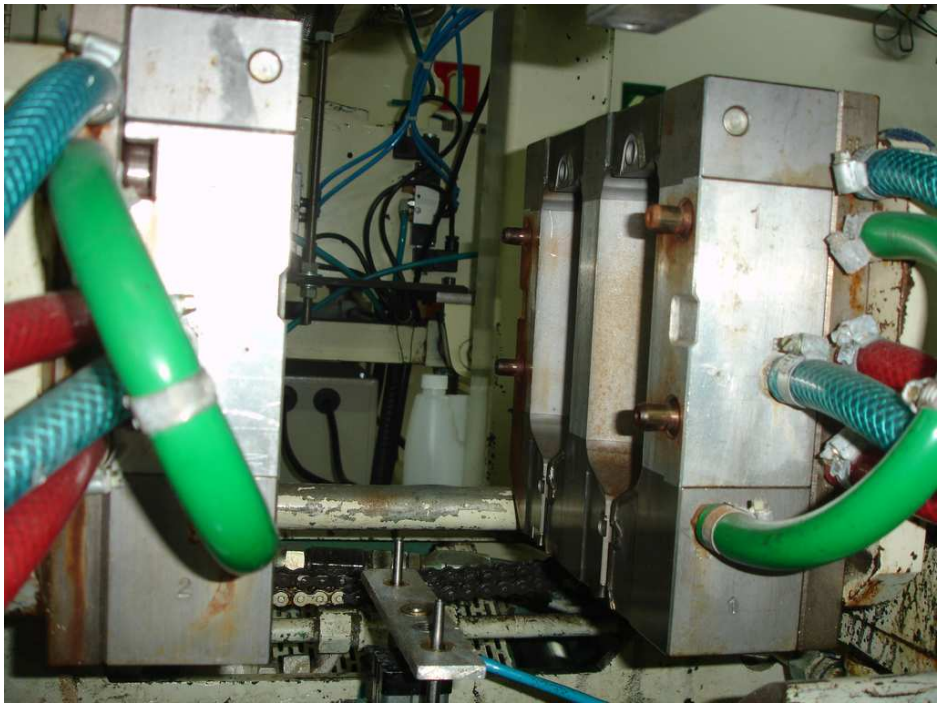

Figura 3.2: Molde utilizado no processo de sopro

Figura 3.3: Processo de sopro

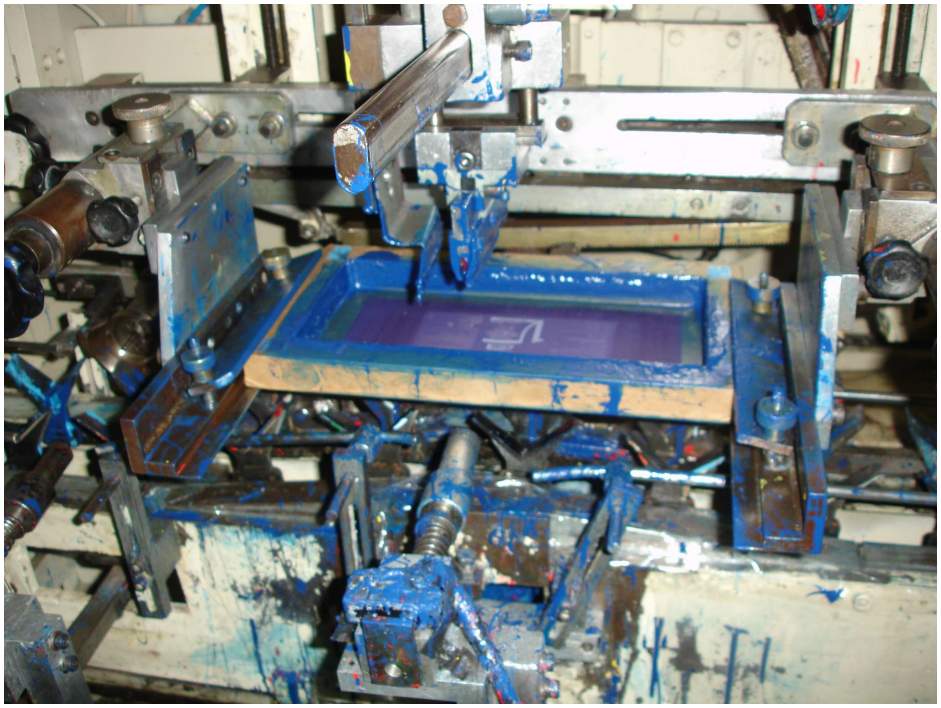

Figura 3.4: Processo de serigrafia

processo consiste na gravação de informações vazando tinta sobre uma tela e pressionando-a através de um rodo ou puxador.

Existe uma grande variedade de produtos no portfólio de produção da empresa. As variações são observadas desde a matéria-prima (utilizando um ou mais materiais), molde (com capacidade variando entre 30ml até 1 litro, além de variados formatos) e no processo de impressão (além de informações personalizadas, os frascos podem receber mais de uma cor na impressão). Ainda assim, alguns produtos finais compartilham o mesmo frasco, o que induz o processo de planejamento da produção a ser dividida entre os dois setores: sopro/injeção e serigrafia.

Diversas restrições operacionais devem ser levadas em consideração para que a programação das máquinas possa ser aplicada na prática.

3.2 Condições do problema

Em casos práticos, não é fácil encontrar na literatura modelos e/ou heurísticas, que possam ser aplicados ao problema, devido a especificidade do processo de planejamento e produção das fábricas. Nestes casos, é preferível uma abordagem que envolve a personalização de modelos e/ou heurísticas.

As particularidades do problema de planejamento da produção da fábrica em questão serão apresentados a seguir.

3.2.1 Data de início de produção

Certas máquinas e/ou tarefas podem ser impossibilitadas de uso desde o início do horizonte, caracterizando assim uma data de início de produção mínimo que deve ser respeitado pela programação da produção.

3.2.2 Data de entrega do produto

A carteira de pedidos da empresa prevê datas de entrega variadas, conforme produto. É necessário, portanto, que o modelo considere esta data de entrega a fim de evitar atrasos e possíveis multas.

3.2.3 Demanda por produtos de setores variados

Os pedidos de compra para a empresa podem envolver produtos com ou sem a impressão de informações no frasco. A programação da produção deve prever a produção de frascos sem a necessidade de utilizar o setor de serigrafia.

3.2.4 Estoque inicial

A produção da fábrica é contínua e, portanto, a programação produção deve considerar o estoque inicial disponível no início do horizonte de planejamento. Como alguns pedidos envolvem os dois setores de produção, este estoque pode envolver um produto intermediário, como um frasco sem impressão.

3.2.5 Produção ininterrupta

Devido ao processo de planejamento, horizonte reduzido e tempos de *setup* elevados, a programação produção deve considerar que a produção de tarefas em má-

quinas deve acontecer sem interrupção e que, portanto, um par tarefa-máquina só deve ter início e fim uma vez ao longo do horizonte de planejamento.

3.2.6 Produção multi-máquinas

A produção de algumas tarefas pode ser realizada em mais de uma máquina, desde que se respeite a produção ininterrupta em cada uma.

3.2.7 Tempo de *setup*

Alguns pares de tarefas, se realizados em sequência numa mesma máquina, possuem tempo de *setup*. Isto se dá devido a possíveis trocas de ferramentas, moldes, materiais, telas de impressão, e outros. O tempo de *setup* para cada par de tarefas é conhecido *a priori*.

3.2.8 Recurso limitado: funcionários

Cada setor possui um número limitado de funcionários disponíveis por turno. Para a operação das máquinas é necessário a alocação de um número variado de funcionários conforme a tarefa a ser realizada. Para que a programação produção tenha aderência com a execução, as máquinas em operação a cada turno não devem exigir um número de funcionários inviável, dado a disponibilidade.

3.2.9 Recurso limitado: materiais

A alimentação das máquinas de sopro/injeção de polietileno e poli-propeno (pequenos grânulos de plástico utilizados para a produção dos frascos e tampas) é feita de maneira automática, através de um sistema de tubos e bombas de sucção. Cada tarefa do setor exige um ou mais materiais. No entanto, o sistema de alimentação possui uma limitação quanto ao número de materiais alimentados a cada turno.

Para o setor de serigrafia, existem ferramentas e materiais que também podem ser recursos escassos e devem ser considerados na programação produção.

3.2.10 Etapas de produção

Pedidos de produtos que exigem a impressão de informações nos frascos geram a necessidade de tarefas em ambos os setores de produção: sopro/injeção (para

a produção do frasco) e serigrafia (impressão). Desta maneira, é necessário que a programação produção considere a ordem destas tarefas e a disponibilidade de estoque a cada turno para o setor de serigrafia, caso contrário, a tarefa de serigrafia não será realizável na prática.

Alguns produtos exigem duas tarefas de serigrafia (casos em que mais de uma cor é impressa no frasco). Nestes casos, também é necessário considerar a ordem e o estoque disponível no início de cada turno para a realização das tarefas dependentes.

3.2.11 Turnos de produção variados

Devido à capacidade de produção mais elevada do setor de serigrafia, a fábrica opera com turnos diferentes em cada um dos setores. Tipicamente, o setor de sopro/injeção opera 24hs em 3 turnos durante a semana e em mais um turno no sábado, enquanto que o setor de serigrafia funciona 8hs por dia (1 turno) ao longo da semana. Devido a estas diferenças, o planejamento deve considerar disponibilidade de turnos e horas diferentes para cada setor.

3.2.12 Possibilidade de hora extra

Mesmo indesejada, existe a possibilidade de operar horas extras nas linhas de produção. O modelo deve considerar esta opção, permitindo, mas penalizando uma solução que utilize horas extras.

3.2.13 Tarefas predecessoras compartilhadas

Alguns produtos usam o mesmo frasco plástico, diferindo apenas na impressão final. Desta maneira, a produção de frascos deve considerar a possibilidade de uso por mais de um produto acabado.

3.3 Classificando o problema

O problema de planejamento de produção da Emplastic não pode ser classificado em uma combinação de triplas conforme apresentado na seção 2. Mesmo assim, o problema compartilha algumas características inclusas na classificação estendida de Pinedo [1995].

3.3.1 Ambiente de máquinas

No problema apresentado, o ambiente de máquinas usado se assemelha ao *flexible job shop* (2.2.1.9), com máquinas não relacionadas (2.2.1.4). Para a produção dos produtos finais, as tarefas devem passar por uma ou ambas as seções da fábrica. No entanto, características como a produção em multi-máquinas e a possibilidade de horas extra, tornam o ambiente de máquinas mais complexo que as classificações disponíveis.

3.3.2 Características de Produção

A maior parte das características de produção apresentadas na seção 2.2.2, são usadas no problema: data de início de produção (rj, 2.2.2.1), tempo de *setup* entre tarefas (s_{j_1,j_2}, 2.2.2.2), preemptividade ($prmp$, 2.2.2.3) porém sem a possibilidade de continuação da produção na mesma máquina, restrições de precedência ($prec$, 2.2.2.4), quebra de máquina ($brkdwn$, 2.2.2.5) através da produtividade zero de uma máquina num turno, restrições máquina-tarefa (M_j, 2.2.2.6), recirculação ($recrc$, 2.2.2.10) através da possibilidade de duas tarefas de produção de um produto final serem de um mesmo setor e *tooling* ($tool$, 2.2.2.11) pelas limitações de uso de materiais e funcionários.

No entanto, características próprias do planejamento e produção da Emplastic impedem a classificação completa do seu processo produtivo. Um exemplo é a possibilidade de produzir-se em um único turno, embalagens para dois produtos que passarão por diferentes processos de serigrafia para serem terminados.

3.3.3 Objetivo

O objetivo do problema da Emplastic é realizar a programação da produção com o menor tempo total de término de todas as tarefa, se enquadrando no objetivo de tempo total de término ponderado ($\sum w_j C_j$, 2.2.3.3), porém com priorização igual para todas as tarefas.

Devido a complexidade do problema, duas restrições foram consideradas *soft constraints*: hora extra e atraso de entrega de produto. Com isto, temos dois novos objetivos que consistem na menor violação possível destas restrições. O segundo objetivo se assemelha ao de menor atraso (L_{max}, 2.2.3.2), enquanto que o primeiro não está previsto na classificação.

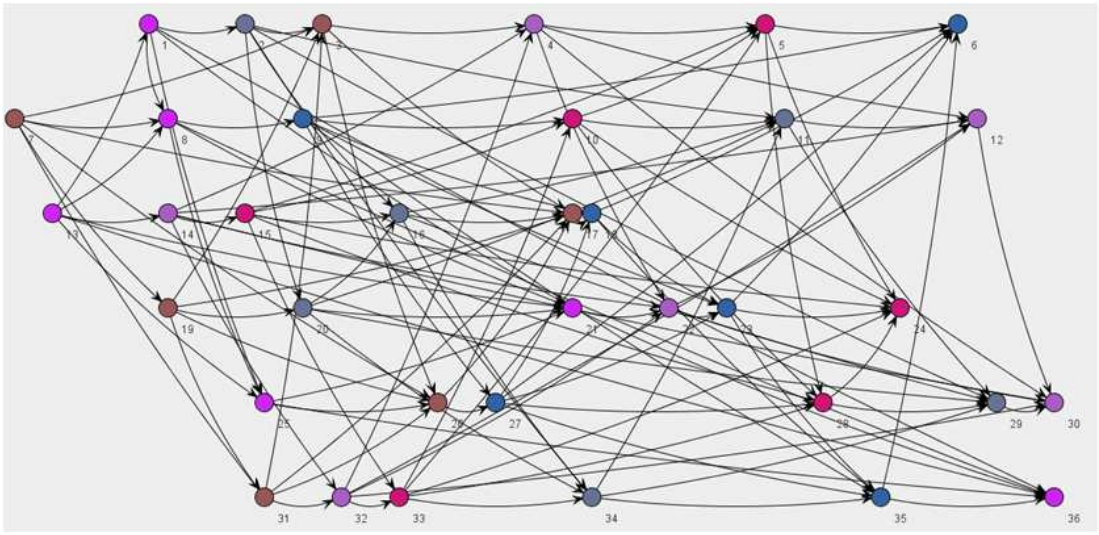

Figura 3.5: Representação do problema de *job shop* na forma de grafo

3.4 Resoluções propostas

Conforme foi apresentado na seção 3.3, não é possível adequar o problema às classificações propostas pela literatura. No entanto, existem diversas abordagens para configurações de problemas que apresentam semelhanças. Em Allahverdi et al. [2008] é possível observar uma pesquisa extensiva de artigos que abordam variações de problemas de esquematização de máquinas com tempos de *setup* ou custos de *setup*.

Para o problema de *job shop* (Jm), muitas abordagens são apresentadas.

Cheung and Zhou [2001] propõem uma abordagem híbrida, entre algoritmo genético e regras heurísticas para o problema com restrições de *setup* e objetivo makespan ($Jm \mid s_{j_1,j_2} \mid C_{max}$);

Ballicu et al. [2002] apresenta um modelo de programação linear inteira mista, que faz uso da representação clássica do problema em um grafo orientado (exemplificada na figura 3.5, onde é apresentado um exemplar do problema FT06, proposto por Fisher and Thompson [1963]) e incluindo na abordagem de tempos de *setup*, restrições de precedência e objetivo de *makespan* mínimo ($Jm \mid s_{j_1,j_2}, prec \mid C_{max}$);

Tahar et al. [2005] apresenta uma abordagem com uma heurística de otimização da colônia de formigas[1] para o problema com máquinas em paralelo para algumas etapas de produção e restrições de tempo de *setup*, tempo de início mínimo e precedência e objetivo de minimizar o tempo total de término $(Jm, Qm \mid s_{j_1,j_2}, r_j, prec \mid \sum C_j)$.

Choi and Choi [2002] também propõem uma abordagem híbrida, com o uso de um modelo de programação linear inteira mista e uma busca local, fazendo uso de algumas propriedades específicas do problema para melhorar o desempenho do algoritmo. Em sua abordagem, Choi and Choi [2002] também considera tarefas que possuem operações alternativas, incluindo características de problemas FJs no problema de $Jm \mid s_{j_1,j_2} \mid C_{max}$.

Muitas abordagens também tratam de *flexible job shop scheduling* (FJm):

Chen et al. [1999] propõe um algoritmo genético para resolver este problema com o objetivo de minimizar o makespan $(FJs \mid\mid C_{max})$.

Gao et al. [2007] trabalha com um algoritmo híbrido, explorando as características de "busca global" do algoritmo genético e as características de "busca local" da heurística *bottleneck shifting*[2]. Neste trabalho temos uma abordagem multi-objetivo $(FJs \mid\mid C_{max}, W_{max}, W_T)$, com o objetivo de minimizar três objetivos simultaneamente: makespan (C_{max}); trabalho total máximo em uma máquina (W_{max}) sendo que este objetivo é utilizado para evitar uma sobrecarga em uma máquina e trabalho total (W_T) para casos em que máquinas diferem em eficiência.

A diversidade de abordagens apresentada na literatura, muitas vezes com a composição de diversas metodologias de resolução diferentes, sugere que a proposta desta dissertação de uma abordagem híbrida, com a utilização de um modelo matemático de programação linear inteira mista, uma heurística de construção e uma heurística de melhoramento, é válida.

---

[1] Também conhecido como ACO, *Ant Colony Optimization*.

[2] Esta heurística consiste em uma busca por novas soluções em diferentes tipos de vizinhanças de uma solução, a partir do caminho crítico obtido pela representação em forma de grafo da solução.

Capítulo 4

Modelo Matemático

4.1 Introdução

Para a resolução deste problema de esquematização de máquinas, um modelo matemático de programação linear inteira mista foi desenvolvido e aplicado em casos reais. As seções apresentadas neste capítulo estão organizadas de maneira a primeiro apresentar os dados (4.2) , as variáveis (4.3), as restrições (4.4) e a função objetivo (4.5) do modelo matemático.

4.2 Parâmetros do Modelo

Uma série de dados de entrada devem ser informados previamente para a definição do modelo. Estes dados definem uma série de parâmetros e características das máquinas, tarefas, fábrica, processo de produção e horizonte de planejamento. São as seguintes a definição e a descrição dos dados:

J Conjunto de tarefas (*jobs*) que devem ser realizadas em máquinas, tais como a produção de um frasco no setor de sopro ou a impressão na embalagem no setor de serigrafia da fábrica. A produção de um produto pode envolver a realização de mais de uma tarefa.

Exemplo
Para determinada programação produção, existe a demanda pelo produto A ao longo do horizonte. A cadeia ou o processo de produção deste produto envolve a geração de um frasco chamado *FA*, uma primeira mão de tinta azul com uma tela e uma segunda mão de tinta verde com uma outra tela, nesta ordem. Neste caso, a realização de 3 tarefas são necessárias para que o produto final A seja completado: *FrascoFA* (geração do frasco FA), *TintaAzulA* (primeira mão de tinta azul) e *TintaVerdeA* (primeira mão de tinta verde).

M Conjunto de máquinas disponíveis na fábrica, tanto do setor de serigrafia, quanto do setor de sopro/injeção.

P Conjunto de períodos disponíveis para o planejamento. Tipicamente caracterizados pelos dias trabalho ao longo de um mês.

T Conjunto de turnos disponíveis ao longo do horizonte de planejamento. O número de turnos pode variar conforme o setor de produção.

S Conjunto de setores da fábrica. Na fábrica em questão, este conjunto é definido por *Sopro* e *Serigrafia*. É importante notar que estamos incorporando as máquinas e tarefas do processo de injeção no setor de sopro, dada a semelhança no processo e planejamento destas tarefas.

R Conjunto de materiais críticos (recurso limitado) utilizados na fábrica.

$Prod_{j,m}$ Produtividade da máquina m para a produção da tarefa j, dado por $unidades/h$.

$Setup_{j_i,j_k}$ Tempo de *setup* para a realização sequêncial das tarefas j_i e j_k dado em horas.

$$JProibido_{j_i,j_k} \begin{cases} 1, & \text{Caso seja proibida a produção simultânea} \\ & \text{das tarefas } j_i \text{e } j_k. \\ 0, & \text{C.c.} \end{cases}$$

Dem_j Demanda pela tarefa j, descontados estoques iniciais da fábrica. A demanda deve ser associada apenas à última tarefa da cadeia de produção do produto demandado.

Exemplo
A partir do exemplo anterior, em que o processo de produção do produto A é apresentado, é definida uma demanda de 100 unidades para este produto. Devemos atribuir esta demanda à última tarefa da cadeia de produção do produto A, neste caso a tarefa $TintaVerdeA$. Logo, $Dem_{TintaVerdeA} = 100$.

$$LigTarefa_{j_i,j_k} \begin{cases} 1, & \text{caso seja necessária a execução da tarefa } j_i, \text{ para} \\ & \text{a posterior execução da tarefa } j_k. \\ 0, & \text{c.c.} \end{cases}$$

A proporção de produção considerada é sempre de 1:1 (um para um).

Exemplo
Continuando o exemplo da produção do produto A, seria necessário estabelecer todas as ligações na cadeia de produção deste produto. Neste caso, temos a sequência $FrascoFA$, $TintaAzulA$ e $TintaVerdeA$ e portanto $LigTarefa_{FrascoA,TintaAzulA} = 1$ e $LigTarefa_{TintaAzulA,TintaVerdeA} = 1$. Desta maneira garantimos que para atender a demanda de 100 unidades estabelecida para a última tarefa da cadeia de produção, 100 tarefas de $TintaAzulA$ já terão sido realizadas, e, por consequência, 100 tarefas de $FrascoFA$ também.

DE_j Turno de entrega da tarefa j.

$FDisp_{t,s}$ Número de funcionários disponíveis na fábrica em cada turno t e cada setor s.

$F_{j,m}$ Número de funcionários necessários para realizar a tarefa j na máquina m.

NM_j Número de moldes disponíveis para a tarefa j.

$$JMat_{j,r} \begin{cases} 1, & \text{caso a tarefa } j \text{ use o material } r. \\ 0, & \text{c.c.} \end{cases}$$

$Matmax$ Número máximo de materiais utilizados num turno.

$JMin_j$ Turno mínimo para início da produção da tarefa j.

$MMin_m$ Turno mínimo para início de utilização da máquina m.

$IMin_{j,m}$ Turno mínimo para a produção da tarefa j na máquina m. Dada pela fórmula: $\max\limits_{j,m}\{MMin_m, JMin_j\}$.

$JSetor_{j,s}$ $\begin{cases} 1, & \text{caso a tarefa } j \text{ pertence ao setor } s. \\ 0, & \text{c.c.} \end{cases}$

JS_j Retorna o setor ao qual a tarefa j pertence.

$HTurno_{t,s}$ número de horas disponíveis para produção no turno t do setor s.

$HExt_{t,s}$ número de horas extras disponíveis para produção no turno t do setor s. Nem todos turnos dispõe de horas extras disponíveis, devido ao início do próximo turno.

$LigPeri_{t_i,t_k}$ $\begin{cases} 1, & \text{caso a produção de uma tarefa predecessora do setor} \\ & \text{sopro no turno } t_i \text{possa ser usada por uma tarefa} \\ & \text{sucessora do setor serigrafia no turno } t_k. \\ 0, & \text{c.c.} \end{cases}$

Dado que os turnos disponíveis nos setores de serigrafia e sopro são diferentes, este parâmetro estabelece as condições para que uma tarefa do setor serigrafia ligada a uma tarefa de sopro tenha estoque suficiente para sua realização.

Exemplo
Se considerarmos 2 turnos por dia para o setor de *sopro* e 1 turno por dia de *serigrafia*, é necessário estabelecer quais turnos de *sopro* podem suprir a demanda do setor de *serigrafia*.

Tabela 4.1: Dias dos turnos do *sopro*

	Turno1	*Turno2*	*Turno3*	*Turno4*	*Turno5*	*Turno6*	*Turno7*
Segunda-feira	x	x					
Terceira-feira			x	x			
Quarta-feira					x	x	
Quinta-feira							x

Tabela 4.2: Dias dos turnos do *serigrafia*

	Turno1	*Turno2*	*Turno3*	*Turno4*
Segunda-feira	x			
Terceira-feira		x		
Quarta-feira			x	
Quinta-feira				x

Considerando que tarefas de *sopro* só podem suprir demandas de tarefas de *serigrafia* se realizadas até o dia anterior, a tabela a seguir apresenta a relação entre os turnos dos setores para o exemplo apresentado.

Tabela 4.3: Relação entre turnos de *sopro* e *serigrafia*

	Turnos de sopro						
Turnos de serigrafia	*Turno1*	*Turno2*	*Turno3*	*Turno4*	*Turno5*	*Turno6*	*Turno7*
Turno1							
Turno2	x	x					
Turno3	x	x	x	x			
Turno4	x	x	x	x	x	x	

Est_j Estoque inicial da tarefa j.

$PeriT_{t,s}$ Retorna o período do turno t no setor s.

4.3 Variáveis

Foram consideradas as seguintes variáveis de decisão pelo modelo:

- $X_{j,m,t} = \begin{cases} 1, & \text{caso a tarefa } j \text{ esteja alocada para a máquina} \\ & m \text{ no turno } t \\ 0, & \text{c.c.} \end{cases}$
- $W_{j,m,t} = \begin{cases} 1, & \text{caso a tarefa } j \text{ iniciou sua produção na máquina} \\ & m \text{ no turno } t \\ 0, & \text{c.c.} \end{cases}$
- $Z_{j,m,t} = \begin{cases} 1, & \text{caso a tarefa } j \text{ tenha finalizada sua produção} \\ & \text{na máquina } m \text{ no turno } t-1 \\ 0, & \text{c.c.} \end{cases}$
- $P_{j,m,t} \geq 0$: Horas de produção da tarefa j na máquina m no turno t.
- $HE_{j,m,t} \geq 0$: Horas extra de produção utilizadas para a tarefa j na máquina m no turno t.
- $Y_j \geq 0$: Término da tarefa j.
- $V_j \geq 0$: Dias de entrega em atraso para a tarefa j.

4.4 Restrições

As restrições que compõem o modelo são listadas abaixo, divididas por categoria:

4.4.1 Definição do período de produção

1. Cada tarefa j pode iniciar sua produção apenas uma vez em cada máquina m:

$$\begin{aligned} &\forall j \in J,\, m \in M: \\ &\sum_{t \in T} W_{j,m,t} \quad \leq \quad 1 \end{aligned} \tag{4.1}$$

2. Cada tarefa j pode finalizar sua produção apenas uma vez em cada máquina m:

$$\begin{aligned} &\forall j \in J,\, m \in M: \\ &\sum_{t \in T} Z_{j,m,t} \quad \leq \quad 1 \end{aligned} \tag{4.2}$$

3. A produção da tarefa j em uma máquina m só pode ser finalizada se a tarefa foi previamente inicializada:

$$\begin{aligned} &\forall j \in J,\, m \in M,\, t_1 \in T: \\ &\sum_{t_2 \in T | t_2 < t_1} W_{j,m,t_2} \quad \geq \quad Z_{j,m,t_1} \end{aligned} \tag{4.3}$$

4. A produção da tarefa j em uma máquina m só pode ser realizada se a tarefa foi previamente inicializada:

$$\begin{aligned} &\forall j \in J,\, m \in M,\, t_1 \in T: \\ &\sum_{t_2 \in T | t_2 \leq t_1} W_{j,m,t_2} \quad \geq \quad X_{j,m,t_1} \end{aligned} \tag{4.4}$$

5. Se a produção da tarefa j em uma máquina m foi iniciada e não finalizada, a tarefa está alocada para a tarefa:

$$\begin{aligned} &\forall j \in J,\, m \in M,\, t_1 \in T: \\ &\sum_{t_2 \in T | t_2 \leq t_1} W_{j,m,t_2} - Z_{j,m,t_2} \quad \leq \quad X_{j,m,t_1} \end{aligned} \tag{4.5}$$

6. A produção da tarefa j em uma máquina m não pode acontecer se a produção foi finalizada:

$$\begin{aligned} &\forall j \in J,\, m \in M,\, t_1 \in T: \\ &1 - \sum_{t_2 \in T | t_2 \leq t_1} Z_{j,m,t_2} \quad \geq \quad X_{j,m,t_1} \end{aligned} \tag{4.6}$$

4.4.2 Atendimento da demanda

1. A demanda associada a cada tarefa j deve ser atendida:

$$\begin{aligned} &\forall j \in J : \\ \sum_{\substack{m \in M \\ t \in T}} (P_{j,m,t} + HE_{j,m.t}) \cdot Prod_{j,m} &\geq Dem_j \end{aligned} \tag{4.7}$$

2. Prazo para produção da tarefa j na máquina m:

$$\begin{aligned} &\forall j \in J : \\ \sum_{\substack{m \in M \\ t \in T | t > DE_j}} X_{j,m,t} &\leq V_j \end{aligned} \tag{4.8}$$

3. O início de produção da tarefa j na máquina m é limitada:

$$\begin{aligned} &\forall j \in J, m \in M : \\ \sum_{t \in T | t < IMin_{j,m}} X_{j,m,t} &\leq 0 \end{aligned} \tag{4.9}$$

4.4.3 Restrições operacionais de produção

1. A máquina m só pode produzir uma tarefa por turno t:

$$\begin{aligned} &\forall m \in M, t \in T : \\ \sum_{j \in J} X_{j,m,t} &\leq 1 \end{aligned} \tag{4.10}$$

2. O número de horas disponíveis para produção em um turno pode ser reduzido devido ao tempo de *setup* da tarefa:

$$\begin{aligned} &\forall j_1, j_2 \in J, m \in M, t \in T : \\ &HTurno_{JS_{j_1},t} - Z_{j_1,m,t} \cdot \min\{Setup_{j_1,j_2}, HTurno_{JS_{j_1},t}\} \\ &\qquad \geq P_{j_2,m,t} \end{aligned} \tag{4.11}$$

3. Limita o número de funcionários sendo utilizados por turno em um dado setor:

$$\begin{aligned} &\forall t \in T,\, s \in S: \\ &\sum_{\substack{j \in J | JSetor_{j,s} \\ m \in M}} X_{j,m,t} \cdot F_{j,m} \;\leq\; FDisp_{t,s} \end{aligned} \tag{4.12}$$

4. Proíbe a produção simultânea de algumas tarefas que não podem ser produzidas no mesmo turno:

$$\begin{aligned} &\forall t \in T: \\ &\forall j_i, j_k \in J \mid JProibido_{j_i,j_k}: \\ &\sum_{m \in M} (X_{j_i,m,t} + X_{j_k,m,t}) \;\leq\; 1 \end{aligned} \tag{4.13}$$

5. Limita o número de moldes para cada tarefa j:

$$\begin{aligned} &\forall j \in J,\, t \in T: \\ &\sum_{m \in M} X_{j,m,t} \;\leq\; NM_j \end{aligned} \tag{4.14}$$

6. Limita o número de materiais sendo usando no turno t:

$$\begin{aligned} &\forall s \in S,\, t \in T: \\ &\sum_{\substack{m \in M \\ r \in R | JMat_{j,r} \\ j \in J | JS_j = s}} X_{j,m,t} \;\leq\; Matmax \end{aligned} \tag{4.15}$$

7. Limita o número de horas de produção por turno:

$$\begin{aligned} &\forall j \in J,\, m \in M,\, t \in T: \\ &X_{j,m,t} \cdot HPeri_{JS_j,t} \;\geq\; P_{j,m,t} \end{aligned} \tag{4.16}$$

8. Limita o número de horas extras praticadas por turno:

$$\begin{aligned} &\forall j \in J,\, m \in M,\, t \in T: \\ &X_{j,m,t} \cdot HExt_{JS_j,t} \;\geq\; HE_{j,m,t} \end{aligned} \tag{4.17}$$

4.4.4 Produção por ligação de precedência

1. Produção devido a ligação entre tarefas de setores diferentes:

$$\forall j \in J,\, t_1 \in T: \sum_{\substack{m \in M \\ t_2 \in T | LigPeri_{t2,t1}}} (P_{j,m,t_2} + HE_{j,m,t_2}) \cdot Prod_{j,m} + Est_j \geq \sum_{\substack{m \in M \\ j_1 \in J | LigTarefa_{j,j_1} \\ t_3 \in T | t_3 \leq t_1}} (P_{j_1,m,t_3} + HE_{j_1,m,t_3}) \cdot Prod_{j,m} \quad (4.18)$$

2. Produção devido a ligação entre tarefas de um mesmo setor:

$$\forall j \in J,\, t_1 \in T: \sum_{\substack{m \in M \\ t_2 \in T | t_2 \leq t_1}} (P_{j,m,t_2} + HE_{j,m,t_2}) \cdot Prod_{j,m} + Est_j \geq \sum_{\substack{m \in M \\ j_1 \in J | LigTarefa_{j,j1} \\ t_3 \in T | t_3 \leq t_1}} (P_{j_1,m,t_3} + HE_{j_1,m,t_3}) \cdot Prod_{j,m} \quad (4.19)$$

4.4.5 Definição do término da tarefa

1. Determina o término de cada tarefa:

$$\forall j \in J,\, m \in M,\, t \in T: \quad Y_j \geq X_{j,m,t} \cdot PeriT_{t,JS_j} \quad (4.20)$$

4.5 Função objetivo

A função objetivo adotada para o modelo matemático é multi-objetivo, e pode ser descrita como:

$$\text{Min } \alpha_1 \cdot \sum_{j \in J} Y_j + \alpha_2 \cdot \sum_{j \in J,\, m \in M,\, t \in T} HE_{j,m,t} + \sum_{j \in J} \alpha_{3,j} \cdot V_j \quad (4.21)$$

Onde α_1 , α_2 e $\alpha_{3,j}$ são definidos de acordo com o objetivo da otimização. Tipicamente, $\alpha_{3,j} \gg \alpha_2 \gg \alpha_1$, assim priorizando, em ordem, a entrega sem atraso, o menor uso possível de horas extras e o menor término total das tarefas.

Capítulo 5

Heurísticas Aplicadas

5.1 Introdução

Muitos problemas práticos e teóricos se resumem à busca de uma solução s pertencente a um conjunto S finito e discreto, tal que s atinga o menor valor possível para um objetivo $f : s \to \mathbb{R}$. Este objetivo, $f(s)$, é dito a função objetivo do problema, o conjunto S o espaço de busca, s uma solução factível e $s*$ uma solução ótima para este problema.

Dentre estes problemas, existe um subconjunto de problemas que são classificados como $NP - Completo$ e que tem como característica um elevado grau de dificuldade de resolução. Para tais problemas, não é conhecido, nem se pode provar a existência ou ausência, de um algoritmo que possa encontrar a solução ótima em tempo polinomial.

No entanto, em muitos destes problemas, não é necessário que se obtenha a melhor solução, apenas uma solução que seja boa, ou próxima o suficiente da melhor, de acordo com algum critério. Para tais problemas é possível utilizar técnicas computacionais de construção ou melhoramento de soluções para encontrar uma solução s', que pode ser uma solução boa o suficiente. Estas técnicas são conhecidas como heurísticas.

Um exemplo bastante conhecido é o problema do caixeiro viajante, ou PCV[1]. Dado n cidades e uma matriz $(n \cdot n)$ de distâncias entre elas, partindo de uma cidade qualquer, qual a sequência de cidades que se deve visitar tal que todas as cidades sejam visitadas apenas uma vez e se volte à cidade de origem com a menor distância possível percorrida.

[1] Também conhecido como *Travelling salesman problem*, ou *TSP*.

Encontrar uma solução viável para este problema é trivial, tal como visitar a cidade 1, depois a cidade 2 e assim por diante até a cidade n e em seguida voltar a cidade 1. No entanto, se considerarmos que a distância de ida é igual à distância de volta entre duas cidades (e, portanto, a matriz de distâncias é simétrica) existem $\frac{(n-1)!}{2}$ sequências possíveis, ou seja, soluções factíveis para o problema.

Enumerar todas as soluções possíveis para este problema para encontrar a melhor seria computacionalmente inviável, pois em um exemplar com 25 cidades existem $3 \cdot 10^{23}$ possíveis sequências. No entanto, é possível aplicar heurísticas para encontrar soluções boas.

5.2 Heurística de construção

Uma possível técnica aplicada a problemas de natureza combinatorial é conhecida como heurística de construção.

Nestes algoritmos busca-se construir uma solução para o problema. Para problemas em que soluções factíveis são triviais, a construção da solução já leva em conta o critério de performance do problema, buscando assim uma solução inicial boa. Para problemas em que soluções factíveis são não-triviais, busca-se, em geral, apenas a construção de uma solução factível.

Heurísticas de construção são usadas em geral como uma etapa inicial no processo de busca de soluções para o problema. A solução inicial construída por este algoritmo serve, em geral, como referência para a próxima etapa do processo de otimização. Esta próxima etapa pode consistir em:

1. Uma heurística de melhoramento: a partir de uma solução inicial, busca-se modificar a mesma com intuito de melhorar a qualidade da mesma, aproveitando-se da estrutura ou características do problema.

2. Uma meta-heurística: Osman and Laporte [1996] definiram formalmente meta-heurísticas como sendo processos de geração iterativos que guiam uma heurística subordinada, combinando inteligentemente técnicas para a busca e exploração do espaço de soluções e usando estratégias de aprendizagem para estruturar informação para a obtenção eficiente de soluções quase ótimas. Em termos práticos, uma meta-heurística estabelece uma estrutura genérica para um algoritmo de resolução de problemas de otimização. Alguns exemplos mais conhecidos são algoritmo genético, busca tabu, GRASP,

recozimento simulado[2] e otimização da colônia de formigas[3]. Uma lista mais extensa pode ser encontrada em Osman and Laporte [1996].

3. Um modelo matemático: para modelos inteiros, encontrar uma solução factível inteira pode ser extremamente difícil. Para alguns *solvers*[4], pode-se utilizar uma solução inicial como ponto de partida na busca por soluções, com intuito de acelerar o processo de otimização. Porém esta técnica não garante um desempenho melhor para o modelo.

Nesta dissertação, serão analisadas as vantagens e desvantagens de utilizar um modelo de programação linear inteira mista com e sem a geração prévia de uma solução inicial através de uma heurística de construção. A utilização ou não de uma heurística de melhoramento buscando melhorias nesta solução inicial também será analisada.

5.3 Aplicação: solução inicial para o problema de esquematização de máquinas

Uma heurística de construção foi construída para gerar uma solução inicial para o problema. Este algoritmo pode ser divido em três partes:

1. Alocação de tarefas com demanda
2. Alocação de tarefas com demanda indireta (com predecessores)
3. Alocação de tarefas com demanda indireta (sem predecessores)

O algoritmo 5.1 apresenta o pseudo-código usado para a construção de solução inicial. A seguir o detalhamento de cada etapa do algoritmo.

---

[2] *Simulated annealing* em inglês.

[3] Também conhecido, em inglês, como *ACO (Ant Colony Optimization)*.

[4] *Solvers* são softwares matemático usados para resolver problemas matemáticos. Muitas vezes estes softwares são usados de maneira complementar a outros software, na forma de uma biblioteca, dll ou chamada externa. Podem ser usados para resolver problemas de programação linear, inteira, não linear, constraint programming, etc.

Algoritmo 5.1 Heurística de Construção

```
/*1a parte*/
/*Alocação de tarefas com demanda direta*/
JA ← CriarConjuntoJA()
ja ← Selecionaja(JA)
Enquanto JA ≠ ∅

        tfa ← Selecionatfa()
        MA ← CriarConjuntoMA()
        tia(MA) ← Selecionatia(MA)
        ma ← Selecionama(tia(MA))
        ModificaSolucao(ja, ma, tia(ma), tfa)
        JA = JA − {ja}
        ja ← Selecionaja(JA)

Fim-enquanto
/*2a parte*/
/*Alocação de tarefas*/
/*com demanda indireta (com predecessor)*/
JB ← CriarConjuntoJB()
jb ← Selecionajb(JB)
Enquanto JB ≠ ∅

        tfb ← Selecionatfb()
        MB ← CriarConjuntoMB()
        tib(MB) ← Selecionatib(MB)
        mb ← Selecionamb(tib(MB))
        ModificaSolucao(jb, mb, tib(mb), tfb)
        JB = JB − {jb}
        JB ← ModificarConjuntoJB(JB)
        jb ← Selecionajb(JB)

Fim-enquanto
/*3a parte*/
/*Alocação de tarefas*/
/*com demanda indireta (sem predecessor)*/
JC ← CriarConjuntoJC()
jc ← Selecionajc(JC)
Enquanto JC ≠ ∅

        tic ← Selecionatic()
        MC ← CriarConjuntoMC()
        tfc(MC) ← Selecionatfc(MC)
        mc ← Selecionamc(tfc(MC))
        ModificaSolucao(jc, mc, tci, tfc(mc))
        JC = JC − {jc}
        jc ← Selecionajc(JC)

Fim-enquanto
```

Algoritmo 5.2 $CriarConjuntoJA()$

```
Função CriarConjuntoJA()
retornar {j ∈ J | Dem_j}
```

Algoritmo 5.3 $Selecionaja(JA)$

```
Função Selecionja(JA)
retornar argmax_{j∈JA} {DE_j}
```

5.3.1 Alocação de tarefas com demanda direta

A primeira fase da heurística de construção trata da alocação de tarefas que possuem demanda direta. Estas tarefas são as últimas da cadeia de produção do produto demandado e seu Dem_j correspondente é maior que zero, conforme definido nos parâmetros do modelo (4.2).

Primeiramente, é preciso criar o conjunto JA que consiste nas tarefas cujas demandas diretas são maiores que zero, conforme o parâmetro Dem_j. Veja o algoritmo 5.2.

Em seguida seleciona-se a primeira tarefa ja a ser alocada. O critério para a seleção da tarefa é a data de entrega, priorizando tarefas que possuem data de entrega mais distantes. Este critério é importante para evitar que alocações sejam sobrepostas conforme forem sendo alocadas. Veja o algoritmo 5.3.

O próximo passo envolve uma estrutura de repetição. Enquanto houver uma tarefa ja selecionada para ser alocada, o loop de alocação deve ser realizado, realizando os seguintes passos:

Definir tfa, o turno de término da tarefa ja. Para tal, busca-se o turno mais próximo ao final do horizonte possível, respeitando uma série de critérios como número máximo de funcionários (3.2.8), tempo de *setup* (3.2.7) e número máximo de materiais (3.2.9), conforme apresentado no algoritmo 5.4.

O conjunto de máquinas MA que poderão ser usadas para produzir ja é definido, conforme definido no algoritmo 5.5.

Dadas as possíveis máquinas (MA), definimos o turno inicial de produção $tia(MA)$ de ja de acordo com cada máquina $m \in MA$. Veja o algoritmo 5.6.

Enfim, seleciona-se a máquina ma, cujo turno de início de produção $tia(MA)$ seja mais próximo ao turno final de produção tfa, para qual a tarefa ja será alocada. Veja o algoritmo 5.7.

Algoritmo 5.4 $Selecionatfa()$

Função $Selecionatfa()$
Selecionar t máximo tal que:
1. O turno deve ser no máximo a data de entrega da tarefa ja ($t \leq DE_{ja}$)
2. Exista máquina m que produza a tarefa ja ($Prod_{ja,m} > 0$) e que:

   a. Esteja disponível no turno ($\nexists j \mid X_{j,m,t} = 1$)
   b. Não possua alocação no próximo turno de tarefa com horas de *setup* de ja ($\nexists j \mid X_{j,m,t+1} \wedge Setup_{ja,j}$)
   c. A alocação da tarefa ja no turno t não ultrapasse o limite de funcionários ($\sum_{j,m'} X_{j,m',t} \cdot F_{j,m'} + F_{ja,m} \leq Fdisp_{t,JS_{ja}}$)

3. A alocação da tarefa ja no turno t não ultrapasse o limite de materiais ($\sum_{\substack{r \in R \mid JMat_{j,r} \\ j \in J \mid JS_{ja} = JS_j \\ m' \in M}} X_{j,m',t} + \sum_{r \in R \mid JMat_{ja,r}} 1 \leq Matmax$)

retornar t

Algoritmo 5.5 $CriarConjuntoMA()$

Função $CriarConjuntoMA()$
Formar conjunto M, com elementos $m \in M$, tal que:

   1. Máquina m produz ja ($Prod_{ja,m}$)
   2. Máquina m esteja disponível no turno tfa ($\nexists j \mid X_{m,j,tfa}$)
   3. Máquina m não possua alocação no próximo turno de tarefa com horas de *setup* de ja ($\nexists j \mid X_{j,m,tfa+1} \wedge Setup_{ja,j}$)

retornar M

Algoritmo 5.6 $Selecionatia(MA)$

Função $Selecionatia(MA)$
Para cada $m \in MA$ selecionar $t(m)$ máximo tal que:

1. A produção de ja na máquina m a partir de $t(m)$ até tfa seja suficiente para atender a demanda de ja:
$$\sum_{t'|t(m)\leq t'\leq tfa} Prod_{ja,m,t'} \cdot HTurno_{t,JS_{ja}} \geq Dem_{ja}$$

retornar $t(MA)$

Algoritmo 5.7 $Selecionama(tia(MA))$

Função $Selecionama(tia(MA))$
retornar $\underset{ma\in MA}{argmax}\{tia(ma)\}$

Dada a tarefa a ser alocada ja, a máquina ma, o turno de início de produção $tia(ma)$ e o turno final de produção tfa, é feita a alocação. Veja o algoritmo 5.8.

Alocada a tarefa ja, remove-se a mesma do conjunto JA, e seleciona-se uma nova tarefa para ser alocada, pelo algoritmo 5.3, reiniciando o loop de alocação.

Ao final desta fase, todas as tarefas com demanda passaram por uma tentativa de alocação. Em casos complexos, é possível que algumas tarefas não sejam alocadas. Nestes casos, a solução inicial não será factível, mas poderá auxiliar o solver do modelo exato a buscar uma solução factível inicial.

5.3.2 Alocação de tarefas com demanda indireta (com predecessores)

Nesta etapa são alocadas as tarefas que possuem demanda indireta, ou seja, que possuí uma demanda que provem da sua ligação de precedência na cadeia de produção ($LigTarefa_{j_1,j_2}$) com uma tarefa já alocada, e que possuem predecessores devido a outra ligação entre tarefas. Estas tarefas são ditas intermediárias, pois possuem predecessores e sucessores.

Inicialmente, criamos o conjunto JB cujos elementos $jb \in JB$ são ligados a alguma tarefa já alocada e que possuem predecessor. Veja o algoritmo 5.9.

`Algoritmo 5.8` $ModificaSolucao(j, m, ti, tf)$

`Função` $ModificaSolucao(j, m, ti, tf)$
`Se` $j \neq \varnothing$ `e` $m \neq \varnothing$ `e` $ti \neq \varnothing$ `e` $tf \neq \varnothing$ `faça`

$W_{j,m,ti} = 1$
$Z_{j,m,tf+1} = 1$
`Para` $t \mid ti \leq t \leq tf$ `faça`

$X_{j,m,t} = 1$
$P_{j,m,t} = HTurno_{t,JS_j}$

`fim-para`

`fim-se`

`Algoritmo 5.9` $CriarConjuntoJB()$

`Função` $CriarConjuntoJB()$

$$DemandaIndireta_j = \sum_{\substack{j' \in J \mid LigTarefa_{j,j'} \\ m \in M \\ t \in T}} P_{j',m,t} \cdot Prod_{j',m} - Est_j$$

`retornar` $\{j \in J \mid DemandaIndireta_j \wedge \exists j' : LigTarefa_{j',j}\}$

Algoritmo 5.10 *Selecionajb*(JB)

`Função` *Selecionajb*(JB)
`retornar` $\underset{jb \in JB}{argmin}\{\min\{PeriT_{t,JS_j}\}\}$
$\underset{\substack{j \in J | LigTarefa_{jb,j} \\ m \in M | X_{j,m,t} \\ t \in T}}{}$

Em seguida selecionamos a primeira tarefa jb a ser alocada nesta etapa. O critério para a escolha busca a tarefa jb cuja tarefa de ligação já alocada é mais próxima ao início do horizonte. Veja o algoritmo 5.10, para maiores detalhes.

O próximo passo envolve uma estrutura de repetição. Enquanto houver uma tarefa jb para alocação repete-se os passos de alocação, realizando os seguintes passos:

Primeiro definimos qual o turno final de produção tfb que será alocado. Para tal, busca-se o turno mais próximo ao final do horizonte possível, respeitando uma série de critérios como número máximo de funcionários (3.2.8), tempo de *setup* (3.2.7), número máximo de materiais (3.2.9) e precedência de tarefas (3.2.10), conforme apresentado no algoritmo 5.11.

O conjunto de máquinas MB que podem produzir jb com término em tfb é criado. Veja o algoritmo 5.12.

Dadas as possíveis máquinas (MB), definimos o turno inicial de produção $tib(MB)$ de jb de acordo com cada máquina $m \in MB$. Veja o algoritmo 5.13.

Enfim, seleciona-se a máquina mb, cujo turno de início de produção $tib(MB)$ seja mais próximo ao turno final de produção tfb, para qual a tarefa jb será alocada. Veja o algoritmo 5.14.

Dadas a tarefa a ser alocada jb, a máquina mb, o turno de início de produção $tib(mb)$ e o turno final de produção tfb, é feita a alocação. Veja o algoritmo 5.8.

Alocada a tarefa jb, remove-se a mesma do conjunto JB, e modifica-se o conjunto JB, em busca de tarefas predecessoras de jb, que possuam predecessores próprios, mas que não haviam sido incluídas ainda no conjunto JB, conforme algoritmo 5.15.

Com o conjunto MB atualizado, seleciona-se uma nova tarefa para ser alocada, pelo algoritmo 5.10 e reinicia-se o loop de alocação.

Ao final desta fase, todas as tarefas com demanda indireta passaram por pelo menos uma tentativa de alocação. Em casos complexos, é possível que algumas tarefas não sejam alocadas. Nestes casos, a solução inicial não será factível, mas poderá auxiliar o solver do modelo exato a buscar uma solução factível inicial.

Algoritmo 5.11 $Selecionatfb()$

Função $Selecionatfb()$
Selecionar t máximo tal que:
1. O turno deve ser no máximo a data de entrega da tarefa jb ($t \leq DE_{jb}$)
2. Exista máquina m que produza a tarefa jb ($Prod_{jb,m}$) e que:

    a. Esteja disponível no turno ($\nexists j \mid X_{j,m,t}$)
    b. Não possua alocação no próximo turno de tarefa com horas de ***setup*** de jb ($\nexists j \mid X_{j,m,t+1} \wedge Setup_{jb,j}$)
    c. A alocação da tarefa jb no turno t não ultrapasse o limite de funcionários ($\sum_{j,m'} X_{j,m',t} \cdot F_{j,m'} + F_{jb,m} \leq Fdisp_{t,JS_{jb}}$)

3. A alocação da tarefa jb no turno t não ultrapasse o limite de materiais ($\sum_{\substack{r \in R \mid JMat_{j,r} \\ j \in J \mid JS_{jb} = JS_j \\ m' \in M}} X_{j,m',t} + \sum_{r \in R \mid JMat_{jb,r}} 1 \leq Matmax$)
4. Não pode existir tarefa sucessora de jb alocada em período anterior ao período de t ($\nexists (j', t', m') \mid X_{j',t'm'} \wedge LigTarefa_{jb,j'} \wedge PeriT_{t',JS_{j'}} < PeriT_{t,JS_{jb}}$)
retornar t

Algoritmo 5.12 $CriarConjuntoMB()$

Função $CriarConjuntoMB()$
Formar conjunto M, com elementos $m \in M$, tal que:

    1. Máquina m produz jb ($Prod_{jb,m}$)
    2. Máquina m esteja disponível no turno tfb ($\nexists j \mid X_{m,j,tfb}$)
    3. Máquina m não possua alocação no próximo turno de tarefa com horas de ***setup*** de jb ($\nexists j \mid X_{j,m,tfb+1} \wedge Setup_{jb,j}$)

retornar M

Algoritmo 5.13 $Selecionatib(MB)$

Função $Selecionatib(MB)$

$$DemandaIndireta_j = \sum_{\substack{j' \in J | LigTarefa_{j,j'} \\ m \in M \\ t \in T}} P_{j',m,t} \cdot Prod_{j',m} - Est_j$$

Para cada $m \in MB$ selecionar $t(m)$ máximo tal que:

1. A produção de jb na máquina m a partir de $t(m)$ até tfb seja suficiente para atender a demanda indireta de jb:

$$\sum_{t' | t(m) \leq t' \leq tfb} Prod_{jb,m,t'} \cdot HTurno_{t,JS_{jb}} \geq DemandaIndireta_j$$

retornar $t(MB)$

Algoritmo 5.14 $Selecionamb(tib(MB))$

Função $Selecionamb(tib(MB))$

retornar $\underset{mb \in MB}{argmax}\{tib(mb)\}$

Algoritmo 5.15 $ModificarConjuntoJB(JB)$

Função $ModificarConjuntoJB(JB)$

Para todo $j \mid LigTarefa_{j,jb}$ faça

$$DemandaIndireta_j = \sum_{\substack{j' \in J | LigTarefa_{j,j'} \\ m \in M \\ t \in T}} P_{j',m,t} \cdot Prod_{j',m} - Est_j$$

retornar $JB \cup \{j \in J \mid DemandaIndireta_j \wedge \exists j' : LigTarefa_{j',j}\}$

Algoritmo 5.16 $CriarConjuntoJC()$

Função $CriarConjuntoJC()$

$$DemandaIndireta_j = \sum_{\substack{j' \in J | LigTarefa_{j,j'} \\ m \in M \\ t \in T}} P_{j',m,t} \cdot Prod_{j',m} - Est_j$$

retornar $\{j \in J \mid DemandaIndireta_j \wedge \nexists j' : LigTarefa_{j',j}\}$

Algoritmo 5.17 $Selecionajc(JC)$

Função $Selecionajc(JC)$

retornar $\underset{jc \in JC}{argmin}\{\min_{\substack{j \in J | LigTarefa_{jc,j} \\ m \in M \\ t \in T | X_{j,m,t}}} \{PeriT_{t,JS_j}\}\}$

5.3.3 Alocação de tarefas com demanda indireta (sem predecessores)

A última etapa consiste na alocação de tarefas que possuem demanda indireta, ou seja, uma demanda que provem da ligação de precedência na cadeia de produção ($LigTarefa_{j_1,j_2}$) com uma tarefa já alocada, e que não possuem predecessores. Estas tarefas são ditas iniciais, pois possuem sucessores, porém não predecessores.

Inicialmente, criamos o conjunto JC cujos elementos $jc \in JC$ são ligados a alguma tarefa já alocada, mas não possuem predecessores. Veja o algoritmo 5.16.

Em seguida selecionamos a primeira tarefa jc a ser alocada nesta etapa. O critério para a escolha busca a tarefa jc cuja tarefa de ligação já alocada é mais próxima ao início do horizonte. Veja o algoritmo 5.17, para maiores detalhes.

O próximo passo envolve uma estrutura de repetição. Enquanto houver uma tarefa jc para alocação repetem-se os passos de alocação, realizando os seguintes passos.

Diferentemente das etapas anteriores, primeiro definimos qual o turno inicial de produção tfi que será alocado. Para tal, busca-se o turno mais próximo ao início do horizonte possível, respeitando uma série de critérios como número máximo de funcionários (3.2.8), tempo de *setup* (3.2.7) e número máximo de materiais (3.2.9), conforme apresentado no algoritmo 5.18.

O conjunto de máquinas MC que podem produzir jc com início em tic é

Algoritmo 5.18 *Selecionatic*()

Função *Selecionatic*()
Selecionar t mínimo tal que:
1. Exista máquina m que produza a tarefa jc ($Prod_{jc,m}$)
e que:

   a. Esteja disponível no turno ($\nexists j \mid X_{j,m,t}$)
   b. Seja pelo menos o turno mínimo de início da tarefa jc na máquina m ($t \geq IMin_{jc,m}$)
   c. A alocação da tarefa jc no turno t não ultrapasse o limite de funcionários ($\sum_{j,m'} X_{j,m',t} \cdot F_{j,m'} + F_{jc,m} \leq Fdisp_{t,JS_{jc}}$)

2. A alocação da tarefa jc no turno t não ultrapasse o limite de materiais ($\sum_{\substack{r \in R \mid JMat_{j,r} \\ j \in J \mid JS_{jc} = JS_j \\ m' \in M}} X_{j,m',t} + \sum_{r \in R \mid JMat_{jc,r}} 1 \leq Matmax$)

retornar t

Algoritmo 5.19 $CriarConjuntoMC()$

Função $CriarConjuntoMC()$
Formar conjunto M, com elementos $m \in M$, tal que:

1. Máquina m produz jc ($Prod_{jc,m}$)
2. Máquina m esteja disponível no turno tic ($\nexists j \mid X_{m,j,tic}$)
3. Tarefa jc na máquina m possa ser utilizada pela data de início mínimo ($tic \geq IMin_{jc,m}$)

retornar M

criado. Veja o algoritmo 5.19.

Dadas as possíveis máquinas (MC), definimos o turno final de produção $tfc(MC)$ de jc de acordo com cada máquina $m \in MC$. Veja o algoritmo 5.20.

Algoritmo 5.20 $Selecionatfc(MC)$

Função $Selecionatfc(MC)$
$$DemandaIndireta_j = \sum_{\substack{j' \in J \mid LigTarefa_{j,j'} \\ m \in M \\ t \in T}} P_{j',m,t} \cdot Prod_{j',m} - Est_j$$
Para cada $m \in MC$ selecionar $t(m)$ mínimo tal que:

1. A produção de jc na máquina m a partir de tic até $tfc(m)$ seja suficiente para atender a demanda indireta de jc:
$$\sum_{t' \mid tic \leq t' \leq tfc(m)} Prod_{jc,m,t'} \cdot HTurno_{t,JS_{jc}} \geq DemandaIndireta_j$$

retornar $t(MC)$

Enfim, seleciona-se a máquina mc, cujo turno de fim de produção $tfc(MC)$ seja mais próximo ao início do horizonte de planejamento. Veja o algoritmo 5.21.

Dadas a tarefa a ser alocada jc, a máquina mc, o turno de início de produção tic e o turno final de produção $tfc(mc)$, é feita a alocação. Veja o algoritmo 5.8.

Alocada a tarefa jc, remove-se a mesma do conjunto JC.

Algoritmo 5.21 $Selecionamc(tfc(MC))$

Função $Selecionamc(tfc(MC))$

retornar $\underset{m \in MC}{argmin}\{tfc(m)\}$

Com o conjunto MC atualizado, seleciona-se uma nova tarefa para ser alocada, pelo algoritmo 5.17, reiniciando o loop de alocação.

Ao final desta fase, todas as tarefas com demanda indireta passaram por pelo menos uma tentativa de alocação. Em casos complexos, é possível que algumas tarefas não sejam alocadas. Nestes casos, a solução inicial não será factível, mas poderá auxiliar o solver do modelo exato a buscar uma solução factível inicial.

5.3.4 Exemplo

Para exemplificar a aplicação da heurística de construção, vamos apresentar seu comportamento num cenário pequeno e ilustrativo.

Consideraremos três produtos A, B e C, com demanda e as seguintes cadeias de produção e tarefas.

- A: $a_1 \rightarrow a_2$
- B: $b_2 \rightarrow b_2$
- C: $c_1 \rightarrow c_2 \rightarrow c_3$

Todas as tarefas são concluídas em 1 turno e podem ser realizadas nas duas máquinas disponíveis: M e N. O horizonte de planejamento contempla 6 turnos (*1*, *2*, *3*, *4*, *5* e *6*).

Seguindo as etapas da heurística, as primeiras tarefas a serem alocadas são as que possuem demanda direta, ou seja, as últimas tarefas de cadeia de produção: a_2, b_2 e c_3. Seguindo a lógica do algoritmo, alocamos a tarefa a_2 na máquina M no turno *6*. Em seguida, alocamos a tarefa b_2 na máquina N no turno *6*. Ao tentarmos alocar a tarefa c_3, é verificado que o turno máximo permitido é o turno *5*. Assim c_3 é alocado na máquina M no turno *5*. A tabela 5.1 ilustra a alocação até este momento do algoritmo.

Na próxima etapa da heurística, vamos alocar tarefas que possuem demanda indireta e que possuem precedentes na cadeia de produção. A tarefa c_2 é a única

Tabela 5.1: Exemplo para heurística de construção: primeira etapa

	Turnos					
Máquina	*1*	*2*	*3*	*4*	*5*	*6*
M					c_3	a_2
N						b_2

que atende a estes critérios, pois possui precedentes (c_1) e possui demanda indireta (devido a alocação da tarefa c_3). Alocamos c_2 na máquina N e turno *4*. A tabela 5.2 ilustra a alocação das tarefas até este momento do algoritmo.

Tabela 5.2: Exemplo para heurística de construção: segunda etapa

	Turnos					
Máquina	*1*	*2*	*3*	*4*	*5*	*6*
M					c_3	a_2
N				c_2		b_2

Por fim, alocamos tarefas com demanda indireta e que não possuem precedentes, ou seja, as tarefas a_1, b_1 e c_1. Alocamos a tarefa a_1 na máquina M no turno *1*, a tarefa b_1 na máquina N e turno *1* e, por fim, a tarefa c_1 na máquina M e turno *2*. A tabela apresenta solução obtida pela heurística.

Tabela 5.3: Exemplo para heurística de construção: terceira etapa

	Turnos					
Máquina	*1*	*2*	*3*	*4*	*5*	*6*
M	a_1	c_1			c_3	a_2
N	b_2			c_2		b_2

5.4 Heurística de melhoramento

Heurísticas de melhoramento consistem em algoritmos cujo objetivo é aperfeiçoar a performance de uma solução previamente definida. Em geral, são complementares às heurísticas de construção servindo como uma segunda etapa no processo de otimização.

As heurísticas de melhoramento diferem das meta-heurísticas por não trabalharem com uma estrutura de busca previamente definida. Por este motivo,

estes algoritmos são, em geral, especializados para um determinado problema e podem usufruir de características próprias deste problema para melhorarem seu desempenho. As heurísticas de melhoramento podem ser utilizadas numa etapa do processo de busca local de uma meta-heurística.

5.5 Aplicação: melhoramento de uma solução de esquematização de máquinas

Uma heurística de melhoramento foi desenvolvida para auxiliar o processo de otimização do problema de esquematização de máquinas abordado nesta dissertação. A sua aplicação deverá ser realizada após a obtenção de uma solução inicial ou até mesmo depois do processo de otimização do modelo de programação matemática, desde que observadas algumas condições. A heurística não faz uso de horas extras e, portanto, uma solução que possuir horas extras poderá ser modificada erroneamente.

O algoritmo visa dois objetivos: melhorar a performance da solução com relação ao critério da função objetivo (uma combinação de minimização do término total das tarefas, somatório dos atrasos e uso de horas extras) e tornar a solução mais factível (de acordo com o critério de número de restrições violadas).

Dado que o problema depende de uma solução inicial, deve-se considerar que as variáveis definidas na seção 4.3 já possuem valores referentes a esta solução e que estas mesmas variáveis serão modificadas para melhoria da solução atual. A estrutura principal da heurística é apresentada no algoritmo 5.22.

5.5.1 Melhoria do término total das tarefas da solução

A primeira fase da heurística de melhoria busca melhorar o término total das tarefas da solução, sem torna a mesma mais infactível.

Primeiramente, devemos criar o conjunto $J1$ que consiste em todas as tarefas que possuem alguma alocação e que deverão ser melhoradas, conforme descrito no algoritmo 5.23.

Uma estrutura de repetição será iniciada para a melhoria das tarefas. Durante cada repetição, um algoritmo tentará realizar esta melhoria enquanto o conjunto $J1$ não for vazio e houver tarefas a serem melhoradas.

Para modificar a solução, primeiro obtém-se uma tarefa (qualquer) $j1$ pertencente a $J1$, a máquina $m1$ ao qual a tarefa está alocada e seu intervalo de alocação definido pelo turno de início $ti1$ e turno de término $tf1$. Veja o algoritmo 5.24.

Algoritmo 5.22 Heurística de Melhoramento

```
/*1a parte*/
/*Melhoria do makespan da solução*/
J1 ← CriarConjuntoCompleto()
Enquanto J1 ≠ ∅

    (j1, m1, ti1, tf1) ← ObtemAlocacao(J1)
    (rbj, rbm, rbti, rbtf, violacoes) ← SalvaAlocacao(j1, m1, ti1, tf1)
    Enquanto ti1 > 1
        HAlocadas ← ObtemHorasAlocadas(j1, ti1, tf1)
        (ti1, tf1) ← AntecipaIntervalo(j1, ti1, tf1, HAlocadas)
        ModificaSolucao(j1, m1, ti1, tf1)
        Se ObtemViolacoes() > violacoes
            ModificaSolucao(rbj, rbm, rbti, rbtf)
            Interrompe-enquanto
        Senão
            (rbj, rbm, rbti, rbtf, violacoes) ←
            SalvaAlocacao(j1, m1, ti1, tf1)
        Fim-se
    Fim-enquanto
    J1 = J1 − {j1}

Fim-enquanto
/*2a parte*/
/*Melhoria na factibilidade da solução*/
J2 ← CriarConjuntoCompleto()
Enquanto J2 ≠ ∅

    (j2, m2, ti2, tf2) ← ObtemAlocacao(J2)
    (rbj, rbm, rbti, rbtf, violacoes) ← SalvaAlocacao(j2, m2, ti2, tf2)
    TFinal ← ObtemTurnoFinal(j2)
    Enquanto ti2 < TFinal
        HAlocadas ← ObtemHorasAlocadas(j2, ti2, tf2)
        (ti2, tf2) ← AdiaIntervalo(j2, ti2, tf2, HAlocadas, TFinal)
        ModificaSolucao(j2, m2, ti2, tf2)
        Se ObtemViolacoes() ≥ violacoes
            ModificaSolucao(rbj, rbm, rbti, rbtf)
            Interrompe-enquanto
        Senão
            (rbj, rbm, rbti, rbtf, violacoes) ←
            SalvaAlocacao(j2, m2, ti2, tf2)
        Fim-se
    Fim-enquanto
    J2 = J2 − {j2}

Fim-enquanto
```

Algoritmo 5.23 *CriarConjuntoCompleto*()

```
Função CriarConjuntoCompleto()
    retornar {j ∈ J | ∃X_{j,m,t} = 1}
```

Algoritmo 5.24 *ObtemAlocacao*(J)

```
Função ObtemAlocacao(J)
rj ←Selecione j ∈ J qualquer
rm ←Selecione m ∈ M tal que ∃t | X_{rj,m,t} = 1
rti ←Selecione o primeiro t tal que X_{rj,rm,t} = 1
rtf ←Selecione o último t tal que X_{rj,rm,t} = 1
    retornar (rj, rm, rti, rtf)
```

No processo de melhoria da solução devemos sempre guardar a solução atualmente alocada, garantindo a possibilidade de retornar a ela caso uma modificação necessite ser desfeita. Para tal, guardamos os valores atuais para a tarefa que será modificada em $(rbj, rbm, rbti, rbtf)$ onde tomamos o prefixo rb do inglês *rollback*, um procedimento conhecido da área de banco de dados na computação e que significa retornar a uma condição anterior às modificações realizadas. Também armazenamos o número de restrições violadas em *violacoes* (veja o algoritmo 5.26), assim será possível verificar se a modificação não trouxe maiores infactibilidades. Veja o algoritmo 5.25.

Devemos iniciar uma nova estrutura de repetição. Um algoritmo que antecipa o intervalo da tarefa $j1$ selecionada será repetido enquanto o início alocado para a tarefa for maior que 1 ($ti1 > 1$), caso contrário não seria possível antecipar mais o intervalo.

Como modificaremos o intervalo de alocação de uma tarefa a uma máquina,

Algoritmo 5.25 *SalvaAlocacao*(j, m, ti, tf)

```
Função SalvaAlocacao(j, m, ti, tf)
Violacoes ← ObtemViolacoes()
    retornar (j, m, ti, tf, Violacoes)
```

Algoritmo 5.26 $ObtemViolacoes()$

```
Função ObtemViolacoes()
retornar número de restrições violadas
na solução atual do problema
```

Algoritmo 5.27 $ObtemHorasAlocadas(j, ti, tf)$

```
Função ObtemHorasAlocadas(j, ti, tf)
retornar  Σ_{t∈T|ti≤t≤tf}  HTurno_{JS_j,t}
```

para evitarmos a geração de novas infactibilidades, devemos armazenar o número de horas atualmente alocadas para o par tarefa/máquina em $HAlocadas$. Veja o algoritmo 5.27.

Definimos um novo intervalo de alocação para a tarefa $j1$ na máquina $m1$, buscando deslocar o intervalo para mais próximo ao início do horizonte. Para tal, devemos antecipar o turno de término em 1 turno e, se for necessário, até que este turno de término possua horas disponíveis (alocação de tarefas para turnos sem horas disponíveis não traz nenhum benefício para a solução). Para completar a alocação, devemos antecipar o turno de início até que o número de horas do intervalo seja pelo menos igual ao número de horas anteriormente alocadas ou que o turno inicial seja o primeiro turno disponível. Veja o algoritmo 5.28.

Modificamos a solução com o novo intervalo $(ti1, tf2)$ para a tarefa $t1$ e máquina $m1$, e usando a mesma função da seção referente a heurística de construção. Veja o algoritmo 5.8.

Em seguida, verifica-se se a modificação da solução piorou a condição de factibilidade do problema, onde $ObtemViolacoes() > violacoes$. Caso esta desigualdade seja verdadeira, devemos reverter a solução para a última versão armazenada nos parâmetros de *rollback* através da chamada *ModificaSolucao(rbj,rbm,rbti,rbtf)*. Também é necessário interromper a estrutura de repetição que visa a antecipação do intervalo da tarefa $j1$, pois será necessário selecionar uma nova tarefa.

No entanto, se a condição não for verdadeira, devemos atualizar nosso *rollback* com a solução atual, já que esta é considerada uma solução melhorada. Veja o algoritmo 5.25.

Finalizamos a estrutura de repetição da antecipação do intervalo de alocação da tarefa $j1$.

Algoritmo 5.28 $AntecipaIntervalo(j, ti, tf, Horas)$

```
Função AntecipaIntervalo(j, ti, tf, Horas)
tf ← tf − 1
Enquanto HTurno_{JS_j,tf} = 0
    tf ← tf − 1
Fim-enquanto
Enquanto ObtemHorasAlocadas(j, ti, tf) < HAlocadas e ti > 1
    ti ← ti − 1
Fim-enquanto
    retornar (ti, tf)
```

Na continuação do algoritmo, devemos remover a tarefa $j1$ do conjunto $J1$, pois esta já não será mais modificada. Com isto, terminamos a estrutura de repetição de melhoria de tarefas e finalizamos a primeira parte do algoritmo.

5.5.2 Melhoria na factibilidade

A segunda parte do algoritmo visa a melhoria na factibilidade do problema, mesmo que isto implique numa piora na função objetivo proposta.

Iniciamos recriando o conjunto $J2$, contendo todas as tarefas com alguma alocação. Veja o algoritmo 5.23.

O próximo passo define uma estrutura de repetição que busca a melhoria na factibilidade do problema, enquanto houver tarefas disponíveis para melhoria ($J2 \neq \varnothing$).

Selecionamos uma tarefa ($j2$) do conjunto $J2$, obtendo a máquina ao qual está alocada ($m2$), seu turno de início ($ti2$) e seu turno de término ($tf2$).

Salvamos a alocação atual da tarefa $j2$ nos parâmetros de *rollback* já definidos e obtemos também a violação atual do problema.

Na segunda parte da heurística de melhoria, iremos adiar o intervalo de alocação da tarefa, então é necessário obter qual o turno máximo de alocação para tarefa. Obtemos e armazenamos este valor em $TFinal$ pela função definida no algoritmo 5.29.

Iniciamos então uma estrutura de repetição que irá adiar o intervalo de alo-

Algoritmo 5.29 $ObtemTurnoFinal(j)$

```
Função ObtemTurnoFinal(j)
retornar max{t ∈ T | HTurno_{JS_j,t} > 0}
```

Algoritmo 5.30 $AdiaIntervalo(j, ti, tf, Horas, TFinal)$

```
Função AdiaIntervalo(j, ti, tf, Horas)
ti ← ti + 1
Enquanto HTurno_{JS_j,ti} = 0
    ti ← ti + 1
Fim-enquanto
Enquanto ObtemHorasAlocadas(j, ti, tf) < HAlocadas e tf <
TFinal
    tf ← tf + 1
Fim-enquanto
    retornar (ti, tf)
```

cação da tarefa selecionada $j2$. Esta repetição se dará enquanto o turno de término da tarefa selecionada não estiver no último turno e puder ser adiada ($tf2 < TFinal$).

Armazenamos o número de horas atualmente alocadas para a tarefa na máquina no parâmetro $HAlocadas$. Veja o algoritmo 5.27.

Modificamos então o intervalo de alocação, adiando-o. Para tal, adiamos em um o turno de início da alocação ($ti2$) e até que haja horas disponíveis neste turno de início. Em seguida, adiamos o turno de término enquanto o número de horas alocadas for menor que o número de horas previamente alocadas e o turno final da alocação ($tf2$) for menor que o último turno disponível. Veja a descrição no algoritmo 5.30.

Modificamos então a solução atual, com o novo intervalo, através da função definida na heurística de construção. Veja o algoritmo 5.8.

Em seguida, verifica-se se a modificação da solução não melhorou a condição de factibilidade do problema, onde $ObtemViolacoes() \geq violacoes$. Caso esta inequação seja verdadeira, devemos reverter a solução para a última versão ar-

mazenada nos parâmetros de *rollback* através da chamada *ModificaSolucao(rbj,rbm,rbti,rbtf)*. Também é necessário interromper a estrutura de repetição que visa adiar o intervalo da tarefa $j2$, pois será necessário selecionar uma nova tarefa.

No entanto, se a condição não for verdadeira, devemos atualizar nosso *rollback* com a solução atual, já que esta é considerada uma solução melhorada. Veja o algoritmo 5.25.

Finalizamos a estrutura de repetição do adiamento do intervalo de alocação da tarefa $j2$.

Na continuação do algoritmo, devemos remover a tarefa $j2$ do conjunto $J2$, pois esta já não será mais modificada. Com isto, terminamos a estrutura de repetição de melhoria de factibilidade e finalizamos a segunda parte do algoritmo.

5.5.3 Exemplo

Para exemplificar a aplicação da heurística de melhoramento, vamos apresentar seu comportamento no cenário descrito na seção 5.3.4, acrescentando uma restrição de que as tarefas dos produtos A e B não podem ser realizadas num mesmo turno.

Partindo a solução obtida pela heurística de construção, vamos aplicar as etapas do algoritmo, começando pela melhoria do término total das tarefas da solução. Seguindo a lógica definida, as seguintes modificações serão realizadas na solução.

1. A tarefa b_2 do turno *6* será realocada para o turno *5*
2. A tarefa c_2 do turno *4* será realocada para o turno *3*
3. A tarefa c_3 do turno *5*será realocada para o turno *4*

A solução obtida por ser observada na tabela 5.4.

Tabela 5.4: Exemplo para heurística de melhoramento: primeira etapa

	Turnos					
Máquina	*1*	*2*	*3*	*4*	*5*	*6*
M	a_1	c_1		c_3		a_2
N	b_2		c_2		b_2	

Na segunda etapa, melhoria na factibilidade, verificamos que a única restrição violada é a de não realização de tarefas dos produtos A e B num mesmo turno, violada pelas tarefas a_1 e b_1. Seguindo a lógica, a tarefa b_1 será realocada do turno *1* para o turno *2*. Não existem mais restrições então não haverá mais modificações, obtendo a solução apresentada na tabela 5.5.

Tabela 5.5: Exemplo para heurística de melhoramento: primeira etapa

	Turnos					
Máquina	*1*	*2*	*3*	*4*	*5*	*6*
M	a_1	c_1		c_3		a_2
N		b_2	c_2		b_2	

Capítulo 6

Resultados Computacionais

6.1 Introdução

Para a realização de testes computacionais do modelo e das heurísticas propostas, cenários reais fornecidos pela empresa Emplastic foram utilizados. Estes cenários foram elaborados a partir dos dados de demanda real, além de estatísticas de produtividades das máquinas. Neste capítulo serão apresentados os três cenários que foram usados para testar o modelo, os resultados obtidos para cada opção de otimização (com heurísticas de construção e melhoramento; apenas com a heurística de construção e sem o uso de heurísticas) e uma análise dos resultados obtidos.

6.2 Cenários

Para avaliação dos métodos de resolução, dados reais foram coletados para a geração de três cenários:

- Emplastic_0809: Cenário referente à programação de máquinas do mês de agosto de 2009, com 30 tarefas, 12 máquinas, 27 períodos e 73 turnos.
- Emplastic_1209: Cenário referente à programação de máquinas do mês de dezembro de 2009, com 40 tarefas, 13 máquinas, 27 períodos e 73 turnos.
- Emplastic_0210: Cenário referente à programação de máquinas do mês de fevereiro de 2010, com 28 tarefas, 11 máquinas, 27 períodos e 81 turnos.

Todos os dados utilizados para a definição destes cenários podem ser encontrados no apêndice A.

6.3 Resultados

Para obter os resultados do modelo e das heurísticas propostas na seção 4 e na seção 5 respectivamente, foi utilizado o sistema AIMMS na versão 3.9.6.13481 com licença de desenvolvedor com *AIMMS size Large*[1] e com solver *CPLEX 12.1 Intermediate*[2]. Os cenários foram resolvidos com as configurações *default* do *CPLEX*, porém, quando foi obtido uma solução inicial pelas heurísticas de construção e melhoramento, a opção *"General - Advanced Start"* foi modificada para *"Use advanced basis"*, opção que obriga o uso da uma solução inicial fornecida. Considerando as expectativas dos planejadores da Emplastic, o tempo limite inicialmente utilizado foi de 1 hora. Não foi estabelecido um limite de GAP para parada.

Conforme descrito na seção 4.5, a função objetivo é composta por 3 componentes: entrega sem atraso, o menor uso possível de horas extras e o menor término total das tarefas realizadas. O peso atribuído ao menor término da última tarefa foi de 0.1; para horas extras realizadas 10; enquanto que entrega sem atraso foi definido por tarefa para cada cenário conforme as tabelas 6.1, 6.2 e 6.3. Dessa maneira, o modelo deverá priorizar encontrar uma solução sem atraso, em seguida, uma solução sem uso de horas extras e, por fim, um de menor término total das tarefas possível.

Tabela 6.1: Emplastic_0809: Penalização de entrega ($\alpha_{3,j}$)

Tarefa	Moldes
Af250P21	10.000
Af50P21	10.000
Bo250P09	10.000
Bo250P10	10.000
Bo250P11	10.000

Continua...

[1] Esta licença permite a geração de cenários de tamanho ilimitado.

[2] Nesta configuração, é permitida a otimização de modelos matemáticos de programação linear inteira mista de tamanho ilimitado.

Tarefa	Moldes
Bo50P09	10.000
Bo50P10	10.000
Bo50P11	10.000
CaEx	100
CPRaLA	100
CPRaLA3,52aCor	1.000
He100	1.000
Po100	1.000
Po250	1.000
TampaUn200	100
TampaUn50	1.000
Un200	100
Va200	1.000
Va500	1.000

Tabela 6.2: Emplastic_1209: Penalização de entrega ($\alpha_{3,j}$)

Tarefa	Moldes
AB500	1.000
Bo250P20	10.000
Bo250P21	10.000
Bo50P20	10.000
Bo50P21	10.000
CaEx	100
CaFu	100
CPRaLA	100
CPRaLA3,52aCor	1.000
MO100	10.000
MO250	1.000
Po100	1.000
Po250	1.000
POONGE125ML	100
PoR100	100
PoR250	1.000

Continua...

Tarefa	Moldes
PoR50	100
PoR502aCor	100
Tr	100
Va200	1.000
Va500	1.000

Tabela 6.3: Emplastic_0210: Penalização de entrega ($\alpha_{3,j}$)

Tarefa	Moldes
AF250P06	10.000
AF50P06	10.000
BO250P03	10.000
BO250P04	10.000
BO250P05	10.000
BO50P03	10.000
BO50P04	10.000
BO50P05	10.000
GL100	100
PO100P01	1.000
PO250P02	1.000
PO250P03	1.000
POBI	100
POBI1	100
POGE125ML	100
POR100P01	100
Tr	100

Os testes foram realizados em um computador portátil com um microprocessador Intel® Core™2 Duo CPU T5250 @ 1.50 GHz com 2 Gb de memória RAM num sistema operacional Windows 7.

Os resultados serão apresentados conforme as configurações de execução utilizadas: *Melhoramento*, *Construção* e *Modelo*. A configuração *Melhoramento* consiste na execução da heurística de construção, seguida da heurística de melhoramento e, por fim, a rodada do modelo matemático pelo *CPLEX 12.1*. Já

a configuração *Construção* consiste na execução da heurística de construção seguida da rodada do modelo pelo *CPLEX 12.1*. Por fim, a configuração *Modelo* consiste apenas na rodada do modelo pelo *CPLEX 12.1*.

6.3.1 Emplastic_0809

O modelo matemático gerado para este cenário possui 10.268 variáveis (sendo 6.435 inteiros), 75.571 restrições e 563.926 não zeros. Foi possível obter soluções factíveis nas três configurações de execução do modelo. A tabela 6.4 descreve os resultados obtidos para cada uma das configurações. Os valores foram obtidos a partir do status final da solução ao término da execução do modelo[3]. O melhor resultado foi obtido a partir da configuração *Melhoramento*, seguido pelo *Modelo* e, por fim, *Construção*. As soluções obtidas não possuem atraso na conclusão das tarefas ou uso de horas extras, sendo o valor da função objetivo integralmente composta pela soma do término total das tarefas.

Tabela 6.4: Emplastic_0809: Resultado obtido

	Configuração		
Item	*Melhoramento*	*Construção*	*Modelo*
Iterações	1.302.883	1.358.102	1.517.642
Total de nós	3.292	5.579	5.898
Nós restantes	1.414	3.630	3.049
Melhor LP	7,71	7,61	7,68
Melhor Solução	20,7	57,2	24,2
GAP	62,7%	86,7%	67,9%
Tempo Total (s)	3.663	3.671	3.667

A tabela 6.5 apresenta o comportamento da rodada de otimização para a configuração *Melhoramento*. A cada nova melhor solução inteira factível[4] encontrada, uma nova linha da tabela foi acrescentada.

[3] Como existe um tempo de execução das heurísticas iniciais, o tempo total de execução pode ultrapassar 3.600 segundos.

[4] Durante o processo de otimização do modelo, o *solver* guarda a melhor solução inteira factível até aquele momento.

Tabela 6.5: Emplastic_0809: Evolução das melhores soluções factíveis para configuração *Melhoramento*

Tempo (s)	Etapa	Lp	MIP	Violações
3	*Construção*	-	55,2	2
137	*Melhoramento*	-	41,5	2
204	*Melhoramento*	-	40,9	2
268	*Melhoramento*	-	40,9	2
2.134	*Solver*	7,27	350,7	-
2.560	*Solver*	7,38	293,9	-
3.010	*Solver*	7,47	22,7	-
3.678	*Solver*	7,61	20,7	-

Ao observar a evolução da otimização na figura 6.1, que apresenta graficamente a evolução da melhor solução inteira ao longo da otimização, é possível verificar que durante o processo de construção e melhoramento da solução inicial não foi possível obter uma solução inicial factível.

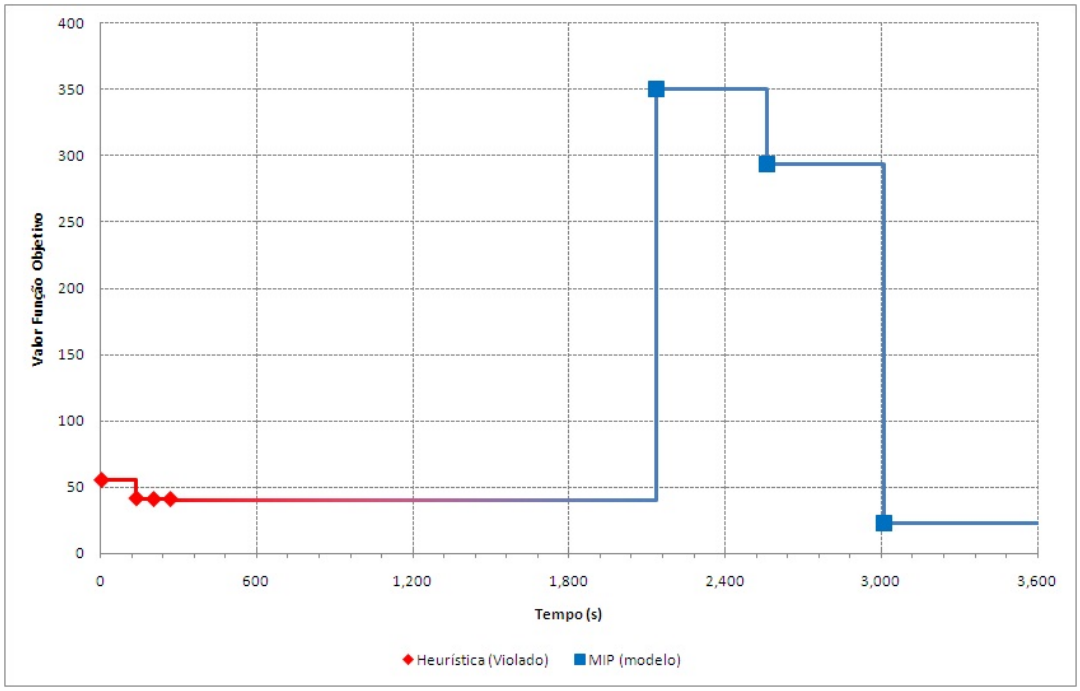

Figura 6.1: Emplastic_0809: Evolução da melhor solução inteira na configuração *Melhoramento*

Evidentemente, o mesmo ocorre para a configuração *Construção*, em que usamos apenas a heurística de construção, conforme tabela 6.6.

Tabela 6.6: Emplastic_0809: Evolução das melhores soluções factíveis para configuração *Construção*

Tempo (s)	Etapa	Lp	MIP	Violações
3	*Construção*	-	55,2	2
1.512	*Solver*	7,37	10.543,1	-
1.650	*Solver*	7,39	116,1	-
2.024	*Solver*	7,41	63,4	-
3.364	*Solver*	7,59	60,6	-
3.450	*Solver*	7,59	60,3	-
3.507	*Solver*	7,60	57,2	-

Mesmo obtendo uma solução inteira factível em menos tempo do que a configuração *Melhoramento*, esta solução apresenta resultado muito inferior e acabamos por não obter uma solução tão boa. Veja a evolução das soluções na figura 6.2.

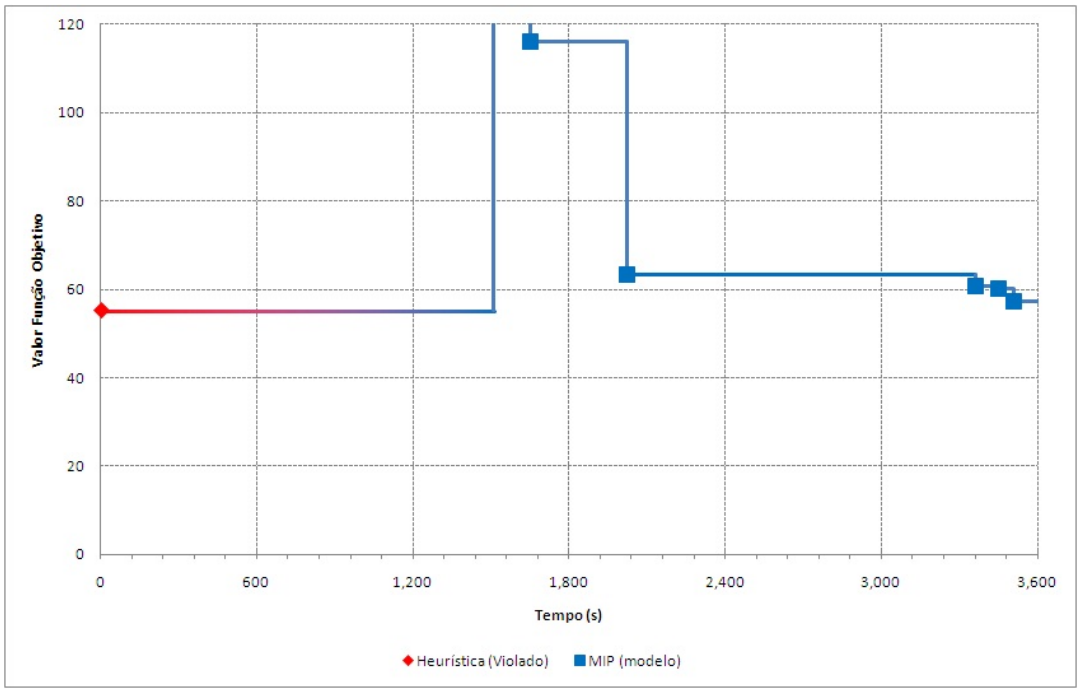

Figura 6.2: Emplastic_0809: Evolução da melhor solução inteira na configuração *Construção*

Por fim, podemos observar as soluções inteiras obtidas com a configuração *Modelo*, em que não usamos nenhuma heurística, na tabela 6.7.

Tabela 6.7: Emplastic_0809: Evolução das melhores soluções factíveis para configuração *Modelo*

Tempo (s)	Etapa	Lp	MIP	Violações
240	*Solver*	5,76	218,4	-
241	*Solver*	5,76	188,2	-
241	*Solver*	5,76	116,5	-
241	*Solver*	5,76	109,0	-
242	*Solver*	5,76	108,8	-
242	*Solver*	5,76	108,4	-
242	*Solver*	5,76	108,3	-
1.147	*Solver*	7,63	100,8	-
1.854	*Solver*	7,68	99,4	-
1.913	*Solver*	7,68	99,3	-
3.380	*Solver*	7,75	94,8	-
3.489	*Solver*	7,75	24,3	-
3.605	*Solver*	7,77	24,2	-

Mesmo sem possuir uma solução inicial para guiar o processo de otimização, esta configuração consegue obter uma solução inteira factível em menor tempo das três opções. Ainda mais, a solução final obtida possui melhor valor que na configuração *Construção*. Veja a evolução das soluções na figura 6.3.

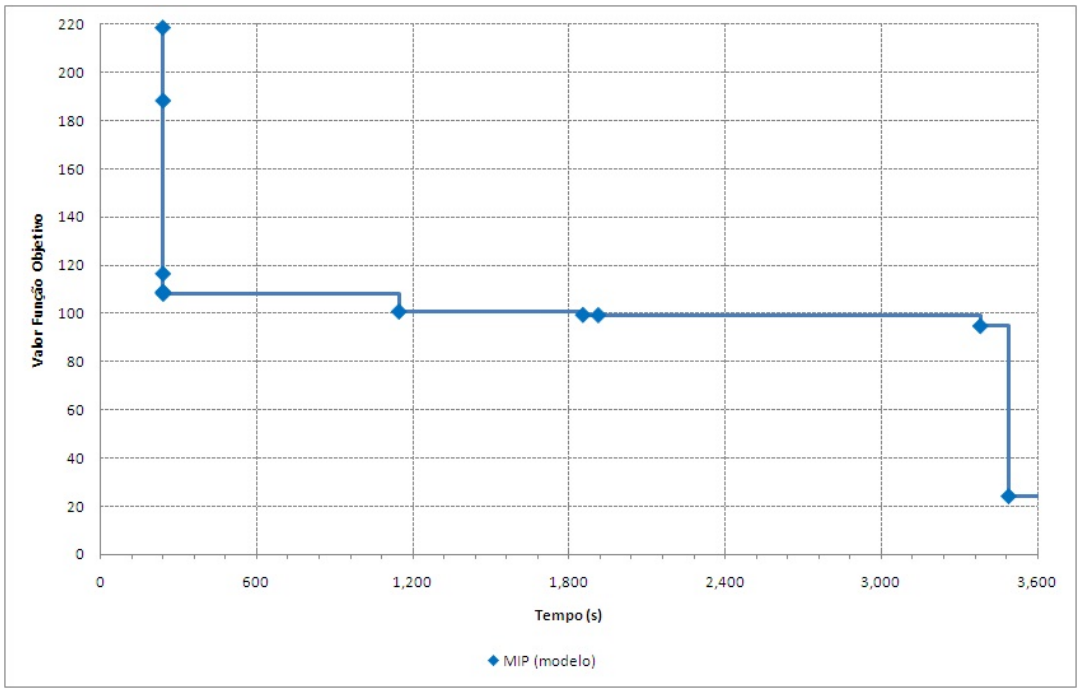

Figura 6.3: Emplastic_0809: Evolução da melhor solução inteira na configuração *Modelo*

6.3.2 Emplastic_1209

O modelo matemático gerado para este cenário possui 12.212 variáveis (sendo 7.977 inteiros), 89.289 restrições e 677.165 não zeros. Dentro do limite de 3.600 segundos, não foi possível obter soluções factíveis nas três configurações de execução do modelo.

Novas rodadas foram realizadas com um limite de tempo estendido de 26.600 segundos (aproximadamente 7 horas e 20 minutos). A tabela 6.8 descreve os resultados obtidos para cada uma das configurações. Assim como no caso anterior, os valores foram obtidos a partir do status final da solução ao término da execução do modelo.

Tabela 6.8: Emplastic_1209: Resultado obtido

	Configuração		
Item	*Melhoramento*	*Construção*	*Modelo*
Iterações	8.066.712	8.359.449	7.828.226
Total de nós	79.285	82.746	59.581
Nós restantes	71.206	49.039	52.875
Melhor LP	110,0	110,0	110,1
Melhor Solução	-	140.7	-
GAP	-	21,9%	-
Tempo Total (s)	27.107	26.705	26.631

É possível observar que apenas a configuração *Construção* obteve uma solução inteira factível. A tabela 6.9 apresenta o comportamento da rodada de otimização para a configuração *Construção*. A cada nova melhor solução inteira factível encontrada, uma nova linha da tabela foi acrescentada. Na última solução factível obtida temos um valor de função objetivo de 140,7. Este valor reflete o atraso de uma tarefa (penalização de 100) e 40,7 a soma do término total de todas as tarefas.

Tabela 6.9: Emplastic_1209: Evolução das melhores soluções factíveis para configuração *Construção*

Tempo (s)	Etapa	Lp	MIP	Violações
1	*Construção*	-	67,4	55
26.384	*Solver*	110,0	157,7	-
26.419	*Solver*	110,0	141,7	-
26.432	*Solver*	110,0	140,7	-

Ao observar a evolução da otimização, também apresentada na figura 6.4, é possível observar que durante o processo de construção e melhoramento da solução inicial não foi possível obter uma solução inicial factível.

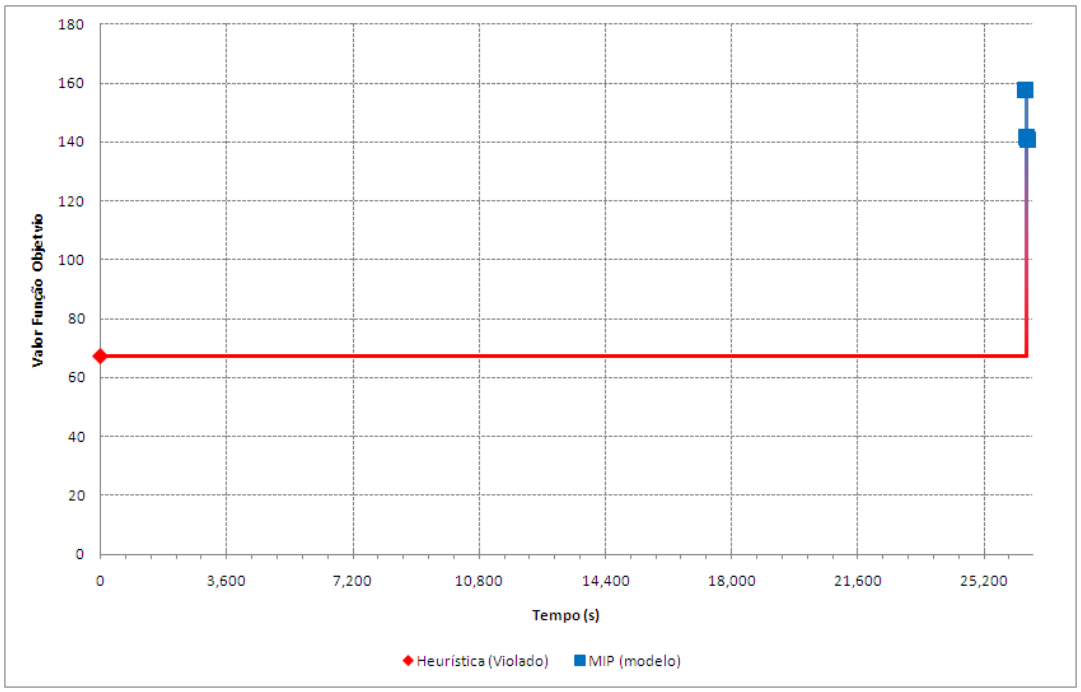

Figura 6.4: Emplastic_1209: Evolução da melhor solução inteira na configuração *Construção*

Este cenário apresentou grande grau de complexidade e o método de otimização apresentou dificuldades para obter soluções factíveis. Ao verificar as heurísticas para obtenção solução inicial, verificamos um número elevado de violações. Estas violações decorrem da impossibilidade de se alocar todas as tarefas pela heurística de construção.

Outra observação relevante a respeito do cenário é que este não possui solução factível sem o uso de horas extras e/ou atraso na entrega de alguma demanda. Esta característica pode ser a causa da complexidade em encontrar uma solução factível.

6.3.3 Emplastic_0210

O modelo matemático gerado para este cenário possui 11.143 variáveis (sendo 6.885 inteiros), 74.138 restrições e 628.905 não zeros. Dentro do limite de 3600 segundos, foi possível obter soluções factíveis nas três configurações de execução do modelo. Características das soluções obtidas para cada uma das configurações

podem ser encontradas na tabela 6.10. Se considerarmos o valor da função objetivo da melhor solução obtida para cada uma das configurações, verificamos que a melhor configuração de execução obtida foi a *Construção*, seguida da *Melhoramento* e *Modelo* por fim. No entanto, considerando o critério de GAP[5] da solução, a configuração *Melhoramento* pode ser considerada a de melhor desempenho, seguida da *Construção* e, por fim, *Modelo*. As soluções obtidas não possuem atraso na conclusão das tarefas ou uso de horas extras, sendo o valor da função objetivo integralmente composta pela soma do término total das tarefas.

Tabela 6.10: Emplastic_0210: Resultado obtido

	Configuração		
Item	*Melhoramento*	*Construção*	*Modelo*
Iterações	1.785.043	1.582.035	1.509.507
Total de nós	14.102	12.388	11.397
Nós restantes	10.919	2.094	583
Melhor LP	8,13	7,44	7,33
Melhor Solução	19,1	17,8	23,5
GAP	57,4%	58,2%	68,8%
Tempo Total (s)	3.904	3.671	3.668

A tabela 6.11 apresenta o comportamento da rodada de otimização para a configuração *Melhoramento*. A cada nova melhor solução inteira factível encontrada, uma nova linha da tabela foi acrescentada.

[5] O GAP é obtido verificando-se qual a diferença relativa da melhor solução atual e a melhor possível solução.

Tabela 6.11: Emplastic_0210: Evolução das melhores soluções factíveis para configuração *Melhoramento*

Tempo (s)	Etapa	Lp	MIP	Violações
3	*Construção*	-	52,7	2
91	*Melhoramento*	-	40,1	0
150	*Melhoramento*	-	35,2	0
194	*Melhoramento*	-	34,7	0
236	*Melhoramento*	-	34,7	0
248	*Solver*	2,56	21,8	-
2470	*Solver*	7,83	20,7	-
3040	*Solver*	7,94	20,3	-
3258	*Solver*	7,98	19,5	-
3649	*Solver*	8,09	19,4	-
3668	*Solver*	8,09	19,3	-
3903	*Solver*	8,13	19,1	-

Ao observar a evolução da otimização na figura 6.5, que apresenta graficamente a evolução da melhor solução inteira ao longo da otimização, é possível verificar que durante o processo de construção e melhoramento da solução inicial foi possível obter uma solução inicial factível. Esta solução factível foi encontrada após a primeira execução da heurística de melhoramento. Logo no início da fase de execução do modelo no *Solver* obtemos uma solução factível, indicando que a utilização de boas heurísticas de construção e melhoramento contribuem para obtenção de soluções factíveis logo no início do processo de otimização.

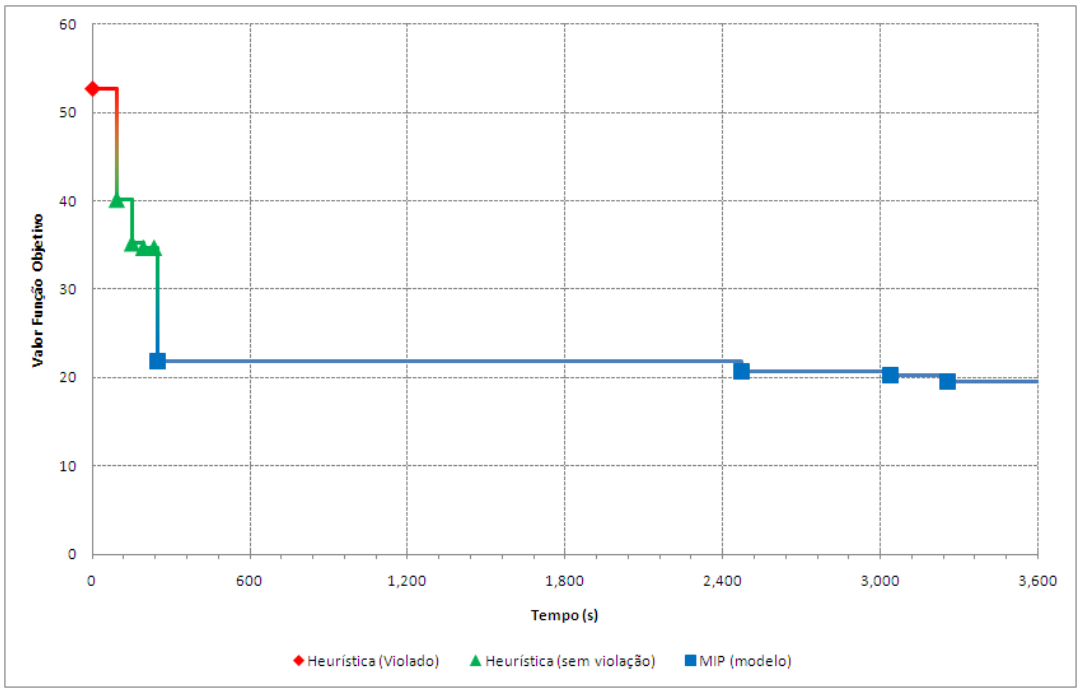

Figura 6.5: Emplastic_0210: Evolução da melhor solução inteira na configuração *Melhoramento*

A tabela 6.12 descreve as soluções obtidas pela configuração *Construção*.

Tabela 6.12: Emplastic_0210: Evolução das melhores soluções factíveis para configuração *Construção*

Tempo (s)	Etapa	Lp	MIP	Violações
2	*Construção*	-	52,7	2
440	*Solver*	6,77	32,9	-
617	*Solver*	6,96	32,8	-
672	*Solver*	6,96	31,8	-
911	*Solver*	7,11	28,5	-
1103	*Solver*	7,18	27,5	-
1471	*Solver*	7,24	26,9	-
1715	*Solver*	7,28	25,2	-
2128	*Solver*	7,32	25,1	-
2561	*Solver*	7,44	23,8	-
2587	*Solver*	7,44	18,9	-
3401	*Solver*	7,44	18,8	-
3586	*Solver*	7,44	18,7	-
3628	*Solver*	7,44	18,6	-
3668	*Solver*	7,44	17,8	-

Ao observar a evolução da otimização na figura 6.6, verificamos que a solução inicialmente obtida pela heurística de construção não é factível. No entanto, é possível observar um bom desempenho do *solver*, eventualmente obtendo a melhor solução factível entre as configurações testadas.

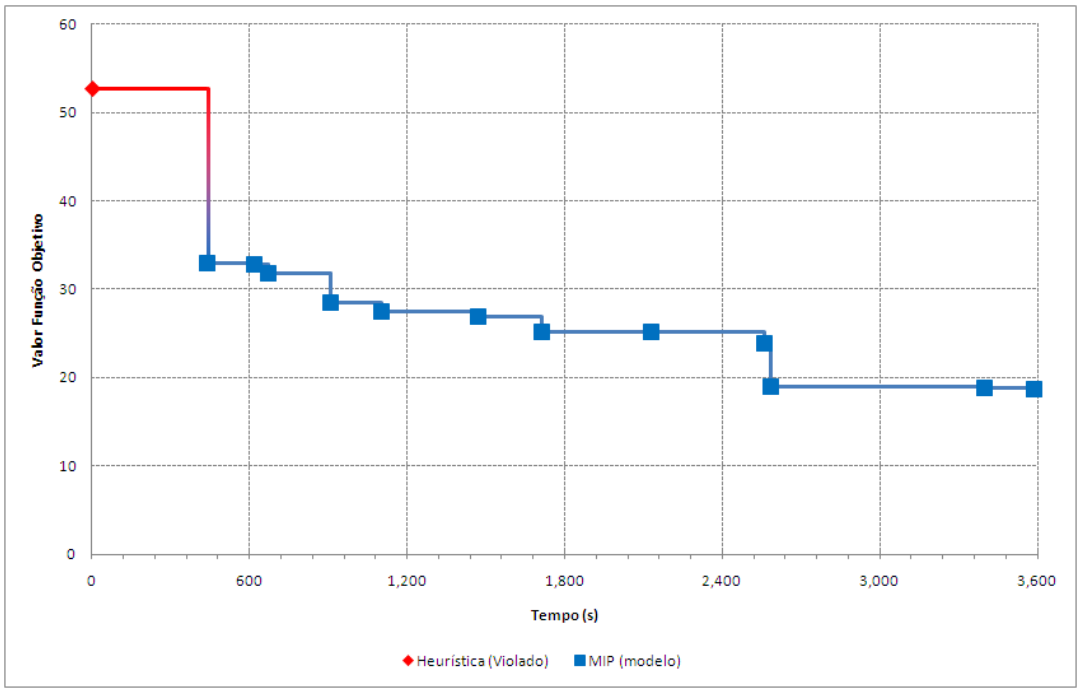

Figura 6.6: Emplastic_0210: Evolução da melhor solução inteira na configuração *Construção*

A tabela 6.13 apresenta o desempenho da configuração *solver*, durante o processo de otimização.

Tabela 6.13: Emplastic_0210: Evolução das melhores soluções factíveis para configuração *Modelo*

Tempo (s)	Etapa	Lp	MIP	Violações
3074	*Solver*	7,33	290.120,1	-
3087	*Solver*	7,33	2.265,0	-
3095	*Solver*	7,33	207,6	-
3338	*Solver*	7,33	140,6	-
3390	*Solver*	7,33	85,7	-
3451	*Solver*	7,33	84,3	-
3493	*Solver*	7,33	82,0	-
3529	*Solver*	7,33	81,7	-
3553	*Solver*	7,33	42,0	-
3603	*Solver*	7,33	25,3	-
3665	*Solver*	7,33	23,5	-

Ao observar a evolução da otimização na figura 6.7, verificamos que nesta configuração houve uma demora maior para o *solver* obter uma solução factível. Ainda assim, esta configuração obteve um resultado bom e apresentou significativas melhorias no final da otimização.

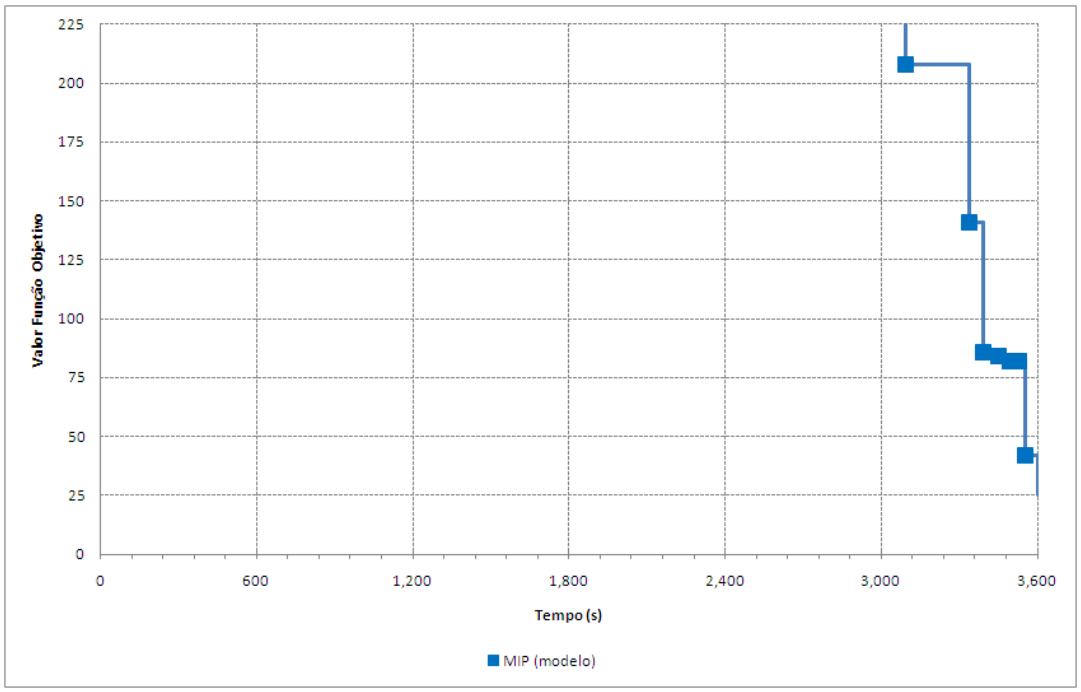

Figura 6.7: Emplastic_0210: Evolução da melhor solução inteira na configuração *Modelo*

6.4 Análise

Os resultados obtidos a partir dos cenários propostos não são conclusivos. Em cada caso analisado, o desempenho das configurações foi variado e não é possível destacar uma melhor configuração.

Contudo, deve-se destacar dois resultados positivos:

- No cenário *Emplastic_1209* apenas ao executar a configuração *Construção* foi possível obter uma solução factível, indicando potencial de auxílio na resolução de cenários complexos.

- No cenário *Emplastic_0210* a configuração *Melhoramento* obteve uma solução factível durante a fase da heurística de melhoramento. Ainda mais, esta solução inicial facilitou a obtenção de uma solução factível durante a execução do *solver*, reduzindo o tempo para a obtenção de uma solução pelo *solver* de 3.074 segundos na configuração *Modelo* para 248 na configuração

Melhoramento, indicando que esta heurística pode auxiliar na resolução de cenários menos complexos e que necessitam de uma solução factível em pouco tempo.

Capítulo 7

Conclusões e trabalhos futuros

Neste trabalho foram descritas características gerais de problemas de esquematização de máquinas, abordando suas muitas variações. Em particular, foi apresentado em detalhe o problema de esquematização da empresa Emplastic, produtora de frascos com impressão.

Não foi possível encontrar na literatura uma proposta de resolução de problemas de esquematização de máquinas que fosse genérica o suficiente para resolver o problema proposto. Como contribuição deste trabalho, foi desenvolvida uma formulação matemática baseada em técnicas de programação linear inteira mista, considerando as muitas restrições particulares ao problema em questão. Para auxiliar nas resolução do modelo matemático, duas heurísticas foram desenvolvidas: uma heurística de construção visando a geração de uma solução factível e uma heurística de melhoramento visando melhorar esta solução inicialmente obtida tornando-a mais factível, além de possíveis melhorias na função objetivo.

Devido ao baixo número de cenários disponíveis, não foi possível verificar qual o potencial de ganho com o uso destas heurísticas.

A abordagem utilizada para resolução do problema de programação de máquinas, apesar de específico, pode servir como base para aplicações em outros problemas.

Para trabalhos futuros, se propõe a aplicação de outras metodologias para resolução deste problema, como meta-heurísticas e heurísticas de melhoramento para aplicação após a execução do *solver*. Uma comparação entre as programações da produção gerados automaticamente pelo sistema desenvolvido neste trabalho e os gerados manualmente pelo planejador da empresa poderia justificar o uso de técnicas de pesquisa operacional para este tipo de trabalho.

Bibliografia

Empresa emplastic, 2010. URL `http://www.emplastic.com.br/empre.html`.

Ali Allahverdi, C. T. Ng, T. C. E. Cheng, and Mikhail Y. Kovalyov. A survey of scheduling problems with setup times or costs. *European Journal of Operations Research*, 187(3):985–1032, June 2008.

M. Ballicu, A. Giua, and C. Seatzu. Job-shop scheduling models with set-up times. *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, 5:95–100, 2002.

Haoxun Chen, Jürgen Ihlow, and Carsten Lehmann. A genetic algorithm for flexible job-shop scheduling. *Proceedings of the 1999 IEEE International Conference on Robotics & Automation*, pages 1120–1125, May 1999.

W Cheung and H. Zhou. Using genetic algorithms and heuristics for job shop scheduling with sequence-dependent setup times. *Annals of Operations Research*, 107:65–81, 2001.

In-Chan Choi and Dae-Sik Choi. A local search algorithm for jobshop scheduling problems with alternative operations and sequence-dependent setups. *Computers & Industrial Engineering*, 42(1):43–58, April 2002.

H. Fisher and G. L. Thompson. *Industrial Scheduling*, chapter Probabilistic learning combinations of local job-shop scheduling rules, pages 225–251. Prentice-Hall, Englewood Cliffs, NJ, 1963.

Henry L. Gantt. *Work, wages, and profits.* Engineering Magazine Co., 1913.

Jie Gao, Mitsuo Gen, Linyan Sun, and Xiaohui Zhao. A hybrid of genetic algorithm and bottleneck shifting for multiobjective flexible job shop scheduling problems. *Computers & Industrial Engineering*, 53:149–162, August 2007.

M. R. Garey, D. S. Johnson, and Ravi Sethi. The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research*, 1(2):117–129, May 1976.

JR Jackson. An extension of johnson's results on job lot scheduling. *Naval Research Logistics Quarterly*, 3:201–203, 1956.

Alan S. Manne. On the job-shop scheduling problem. *Operations Research*, 8(2): 219–223, Mar. - Apr. 1960.

Ibrahim H. Osman and Gilbert Laporte. Metaheuristics: A bibliography. *Annals of Operations Research*, 63:513–623, 1996.

Michael Pinedo. *Scheduling: theory, algorithms, and systems*. Englewood Cliffs, Prentice Hall, 1995.

Dominick V. Rosato, Donald V. Rosato, and Marlene G. Rosato. *Injection Molding Handbook*. Springer, 3rd edition, October 2000.

D.N. Tahar, F. Yalaoui, L. Amodeo, and C Chu. An ant colony system minimizing total tardiness for hybrid job shop scheduling problem with sequence dependent setup times and release dates. *Proceedings of the International Conference on Industrial Engineering and Systems Management*, pages 469–478, May 2005.

Harvey M. Wagner. An integer linear-programming model for machine scheduling. *Naval Research Logistics Quarterly*, 6(2):131–140, 1959.

M. Widmer. Job shop scheduling with tooling constraints: A tabu search approach. *The Journal of the Operational Research Society*, 42(1):75–82, Jan. 1991.

Apêndice A

Cenários

A seguir serão apresentados todos os dados referentes a cada um dos cenários analisados. A descrição dos parâmetros pode ser encontrada na seção 4.2.

A.1 Emplastic_0809

Tarefas (J): Af250P21, Af50P21, Bo250P09, Bo250P10, Bo250P11, Bo50P09, Bo50P10, Bo50P11, CaEx, CPRaLA, 'CPRaLA3,5', 'CPRaLA3,52aCor', FrsAf250P21, FrsAf50P21, FrsCPRALA, 'FrsCPRALA3,5', FrsHe100, FrsPo250, FrsUn200, He100, I250NPPHCP, I500NPPH, I50NPPHCP, Po100, Po250, TampaUn200, TampaUn50, Un200, Va200, Va500

Máquinas (M): BPS, INJ1, INJ2, PZ, RG1, RG10, RG3, RG4, RG5, RG6, 'Silk 1', 'Silk 2', 'Silk 3', 'Silk 5', 'Silk 6', 'Silk 7'

Períodos (P): 0..26

Turnos (T): 1..73

Setores (S): Sopro, Serigrafia

Materiais (R): PPHCP, PEAD, PEBD, PPH, BPEAD

Tabela A.1: Emplastic_0809: Produtividade ($Prod_{j,m}$) [unidades/h]

Máquina	Tarefa	Produtividade
INJ1	*TampaUn200*	360
INJ2	*TampaUn50*	1.080
RG10	*FrsAf250P21*	982
	FrsPo250	982
	I250NPPHCP	982
RG4	*FrsUn200*	400
	FrsCPRALA	328
	FrsCPRALA3,5	328
	I500NPPH	424
RG5	*FrsAf50P21*	1.847
	I50NPPHCP	1.800
RG6	*FrsHe100*	1.440
Silk 1	*Bo50P09*	3.000
	Af50P21	3.000
	Af250P21	3.000
	Po250	1.500
	Bo250P09	3.000
	Bo250P10	3.000
	Bo50P10	3.000
	Bo50P11	3.000
	Bo250P11	3.000
Silk 2	*CaEx*	1.020
	Va200	2.400
	Va500	3.000
	Af250P21	3.000
	Po250	1.500
	Bo250P09	3.000
	Bo250P10	3.000
	Bo250P11	3.000
Silk 3	*Po100*	2.400
	Po250	2.400
Silk 5	*CPRaLA3,5*	2.400

Continua...

Máquina	Tarefa	Produtividade
	Bo50P09	3.000
	Po100	1.500
	He100	3.000
	Va200	3.000
	Af50P21	3.000
	Af250P21	3.000
	Bo250P09	3.000
	Un200	1.200
	Bo250P10	3.000
	Bo50P10	3.000
	Bo50P11	3.000
	Bo250P11	3.000
	CPRaLA	2.400
	CPRaLA3,52aCor	2.400
Silk 6	*Bo50P09*	3.000
	Po100	1.500
	He100	3.000
	Va200	3.000
	Af50P21	3.000
	Af250P21	3.000
	Bo50P10	3.000
	Bo50P11	3.000
	CPRaLA3,52aCor	2.400
Silk 7	*Bo50P09*	4.800
	Af50P21	4.800
	Bo50P10	4.800
	Bo50P11	4.800

Tabela A.2: Emplastic_0809: Tempo de *setup* de tarefas do setor de serigrafia ($Setup_{j_i,j_k}$) [h]

	Tarefa Sucessora																	
Tarefa Predecessora	*CPRaLA3,5*	*Bo50P09*	*CaEx*	*Po100*	*He100*	*Va200*	*Va500*	*Af50P21*	*Af250P21*	*Po250*	*Bo250P09*	*Un200*	*Bo250P10*	*Bo50P10*	*Bo50P11*	*Bo250P11*	*CPRaLA*	*CPRaLA3,52aCor*
CPRaLA3,5		8	8	8	8	8		8	8	8	8	8	8	8	8	8	2	2
Bo50P09	8		8	8	8	8		2	8	8	8	8	8	2	2	8	8	8
CaEx	8	8		8	8	8	2	8	8	8	8	8	8	8	8	8	8	8
Po100	8	8	8			8		8	8	8	8	8	8	8	8	8	8	8
He100	8	8	8			8		8	8		8	8	8	8	8	8	8	8
Va200	8	8	8	8	8		8	8	8	8	8	8	8	8	8	8	8	8
Va500	8	8	2	8	8	8			8	8	8	8	8	8	8	8	8	8
Af50P21	8	2	8	8	8	8	8		8	8	8	8	8	2	2	8	8	8
Af250P21	8	8	8	8	8	8	8	8		2	2	8	2	8	8	2	8	8
Po250	8	8	8	8		8	8	8	2		2		2	8	8	2	8	8
Bo250P09	8	8	8	8	8	8	8	8	2	2		8	2	8	8	2	8	8
Un200	8	8	8	8	8	8	8	8	8	8	8		8	8	8	8	8	8
Bo250P10	8	8	8	8	8	8	8	8	2	2	2	8		8	8	2	8	8
Bo50P10	8	2	8	8	8	8		2	8	8	8	8	8		2	8	8	8
Bo50P11	8	2	8	8	8	8		2	8	8	8	8	8	2		8	8	8
Bo250P11	8	8	8	8	8	8	8	8	2	2	2	8	2	8	8		8	8
CPRaLA	2	8	8	8	8	8		8	8	8	8	8	8	8	8	8		2
CPRaLA3,52aCor	2	8	8	8	8	8		8	8	8	8	8	8	8	8	8	2	

Tabela A.3: Emplastic_0809: Tempo de *setup* de tarefas do setor de sopro ($Setup_{j_i,j_k}$) [h]

	Tarefa Sucessora										
Tarefa Predecessora	*TampaUn50*	*FrsAf50P21*	*FrsAf250P21*	*FrsPo250*	*TampaUn200*	*FrsUn200*	*FrsCPRALA*	*FrsCPRALA3,5*	*I50NPPHCP*	*I250NPPHCP*	*I500NPPH*
TampaUn50					8						
FrsAf50P21				2					1		
FrsAf250P21				2						1	
FrsPo250			2							2	
TampaUn200	8										
FrsUn200							8	8			8
FrsCPRALA						8		1			8
FrsCPRALA3,5						8	1				8
I50NPPHCP		1									
I250NPPHCP			1	2							
I500NPPH						8	8	8			

Tabela A.4: Emplastic_0809: Demanda (Dem_j) [unidades]

Tarefa	Demanda
Af250P21	46.500
Af50P21	85.000
Bo250P09	36.000
Bo250P10	36.000
Bo250P11	36.000
Bo50P09	120.000
Bo50P10	120.000
Bo50P11	120.000
CaEx	1.500
CPRaLA	15.000
CPRaLA3,52aCor	15.000

Continua...

Tarefa	Demanda
He100	50.000
Po100	50.000
Po250	20.000
TampaUn200	23.040
TampaUn50	29.000
Un200	23.040
Va200	42.000
Va500	20.000

Tabela A.5: Emplastic_0809: Ligação de tarefas ($LigTarefa_{j_i,j_k}$)

Tarefa Predecessora	Tarefa Sucessora
CPRaLA3,5	*CPRaLA3,52aCor*
FrsAf250P21	*Af250P21*
FrsAf50P21	*Af50P21*
FrsCPRALA	*CPRaLA*
FrsCPRALA3,5	*CPRaLA3,5*
FrsHe100	*He100*
FrsPo250	*Po250*
FrsUn200	*Un200*
I250NPPHCP	*Bo250P09*
	Bo250P10
	Bo250P11
I500NPPH	*CaEx*
	Va500
I50NPPHCP	*Bo50P09*
	Bo50P10
	Bo50P11

Tabela A.6: Emplastic_0809: Data entrega (DE_j)

Tarefa	Turno
Af250P21	*13*

Continua...

Tarefa	Turno
Af50P21	*13*
Bo250P09	*17*
Bo250P10	*23*
Bo250P11	*27*
Bo50P09	*17*
Bo50P10	*23*
Bo50P11	*27*
CaEx	*8*
CPRaLA	*19*
CPRaLA3,52aCor	*19*
He100	*12*
Po250	*16*
TampaUn200	*53*
TampaUn50	*16*
Un200	*19*
Va200	*12*
Va500	*12*

Máximo funcionários ($FDisp_{t,s}$): $\begin{cases} 'Sopro': & 5 \\ 'Serigrafia': & 15 \end{cases}, \forall t$

Tabela A.7: Emplastic_0809: Funcionários para qualquer tarefa nas máquinas ($F_{j,m}$)

Máquina	Funcionários
BPS	1
INJ2	1
RG10	1
RG4	1
RG5	1
RG6	1
Silk 1	3
Silk 2	3
Silk 3	3

Continua...

Máquina	Funcionários
Silk 5	3
Silk 6	3
Silk 7	3

Tabela A.8: Emplastic_0809: Moldes disponíveis (NM_j)

Tarefa	Moldes
Af250P21	4
Af50P21	4
Bo250P09	3
Bo250P10	3
Bo250P11	3
Bo50P09	4
Bo50P10	4
Bo50P11	4
CaEx	1
CPRaLA	1
CPRaLA3,5	1
CPRaLA3,52aCor	2
FrsAf250P21	1
FrsAf50P21	1
FrsCPRALA	1
FrsCPRALA3,5	1
FrsHe100	1
FrsPo250	1
FrsUn200	1
He100	2
I250NPPHCP	1
I500NPPH	1
I50NPPHCP	1
Po100	3
Po250	3
TampaUn200	1
TampaUn50	1
Un200	1

Continua...

Tarefa	Moldes
Va200	3
Va500	1

Tabela A.9: Emplastic_0809: Tarefa usa material ($JMat_{j,r}$)

	Material		
Tarefa	*PPHCP*	*PEAD*	*PPH*
FrsAf250P21	x		
FrsAf50P21	x		
FrsCPRALA		x	
FrsCPRALA3,5		x	
FrsHe100		x	
FrsPo250		x	
I250NPPHCP	x		
I500NPPH			x
I50NPPHCP	x		

Máximo materiais ($Matmax$): 4

Início mínimo tarefa ($JMin_j$): Todas as tarefas podem iniciar no primeiro turno.

Início mínimo máquina ($MMin_m$): Todas as máquinas podem começar no primeiro turno.

Tabela A.10: Emplastic_0809: Setor de tarefa ($JSetor_{j,s}$)

	Setor	
Tarefa	*Serigrafia*	*Sopro*
Af250P21	x	
Af50P21	x	
Bo250P09	x	
Bo250P10	x	
Bo250P11	x	

Continua...

	Setor	
Tarefa	*Serigrafia*	*Sopro*
Bo50P09	x	
Bo50P10	x	
Bo50P11	x	
CaEx	x	
CPRaLA	x	
CPRaLA3,5	x	
CPRaLA3,52aCor	x	
FrsAf250P21		x
FrsAf50P21		x
FrsCPRALA		x
FrsCPRALA3,5		x
FrsHe100		x
FrsPo250		x
FrsUn200		x
He100	x	
I250NPPHCP		x
I500NPPH		x
I50NPPHCP		x
Po100	x	
Po250	x	
TampaUn200		x
TampaUn50		x
Un200	x	
Va200	x	
Va500	x	

Tabela A.11: Emplastic_0809: Horas normais e extras disponíveis por semana na serigrafia ($HTurno_{t,s}$ e $HExt_{t,s}$) [h]

	Segunda	*Terça*	*Quarta*	*Quinta*	*Sexta*	*Sábado*	*Domingo*
Normal	8	8	8	8	8	0	0
Extra	2	2	2	2	2	8	0

Tabela A.12: Emplastic_0809: Horas normais e extras disponíveis por semana no setor sopro ($HTurno_{t,s}$ e $HExt_{t,s}$) [h]

	1o Turno (Seg)	*2o Turno (Seg)*	*3o Turno (Seg)*	*1o Turno (Ter)*	*2o Turno (Ter)*	*3o Turno (Ter)*	*1o Turno (Qua)*	*2o Turno (Qua)*	*3o Turno (Qua)*	*1o Turno (Qui)*	*2o Turno (Qui)*	*3o Turno (Qui)*	*1o Turno (Sex)*	*2o Turno (Sex)*	*3o Turno (Sex)*	*1o Turno (Sab)*	*2o Turno (Sab)*	*3o Turno (Sab)*	*Domingo*
Normal	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	5	5	5	0
Extra	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	8

Dia do período inicial: Terça-feira

Tabela A.13: Emplastic_0809: Estoque inicial (Est_j) [unidades]

Tarefa	Estoque
FrsHe100	50.000
FrsPo250	20.000

A.2 Emplastic_1209

Tarefas (J): AB500, Bo250P20, Bo250P21, Bo50P20, Bo50P21, CaEx, CaFu, 'CPRaLA3,5', 'CPRaLA3,52aCor', CPRaLA, FrBo250P21, FrBo50P20,

FrBo50P21, FrCaEx, FrCaFu, FrPoR100, FrTr, I100BPPH, I100NPEAD, I200NPEAD, I250BPPH, I250NPEAD, I250NPEADPoR, I250NPPHCP, I500NPPH, I50NPEAD, I50NPPHCP, MO100, MO250, P500MPEAD, P500UPEAD, Po100, Po250, PO250NPEAD, POONGE125ML, PoR100, PoR250, PoR50, PoR502aCor, Tr, Va200, Va500

Máquinas (M): BPS, INJ1, INJ2, PZ, RG1, RG10, RG3, RG4, RG5, RG6, 'Silk 1', 'Silk 2', 'Silk 3', 'Silk 5', 'Silk 6', 'Silk 7', 'Silk 8'

Períodos (P): 0..26

Turnos (T): 1..73

Setores (S): Sopro, Serigrafia

Materiais (R): PPHCP, PEAD, PEBD, PPH, BPEAD

Tabela A.14: Emplastic_1209: Produtividade ($Prod_{j,m}$) [unidades/h]

Máquina	Tarefa	Produtividade
BPS	*FrTr*	772
	I200NPEAD	720
PZ	*AB500*	600
RG10	*FrBo250P21*	982
	I250BPPH	982
	I250NPEAD	982
	I250NPEADPoR	982
	I250NPPHCP	982
RG4	*FrCaEx*	424
	FrCaFu	424
	FrTr	772
	I500NPPH	424
	P500MPEAD	328
	P500UPEAD	328
	PO250NPEAD	240

Continua...

Máquina	Tarefa	Produtividade
RG5	*FrBo50P20*	1.800
	FrBo50P21	1.800
	I50NPEAD	1.800
	I50NPPHCP	1.800
RG6	*FrPoR100*	1.440
	I100BPPH	1.310
	I100NPEAD	1.440
Silk 1	*Bo250P20*	3.000
	Bo250P21	3.000
	Bo50P20	3.000
	Bo50P21	3.000
	Po250	1.500
Silk 2	*Bo250P20*	3.000
	Bo250P21	3.000
	CaEx	1.020
	CaFu	1.020
	Po250	1.500
	Va200	2.400
	Va500	3.000
Silk 3	*Po100*	2.400
	Po250	2.400
	PoR100	2.400
	PoR250	2.400
	Tr	2.100
Silk 5	*Bo250P20*	3.000
	Bo250P21	3.000
	Bo50P20	3.000
	Bo50P21	3.000
	CPRaLA3,5	2.400
	CPRaLA3,52aCor	2.400
	CPRaLA	2.400
	MO100	3.000
	MO250	3.000
	Po100	1.500
	PoR50	3.000

Continua...

Máquina	Tarefa	Produtividade
	Va200	3.000
Silk 6	*Bo50P20*	3.000
	Bo50P21	3.000
	CPRaLA3,52aCor	2.400
	MO250	3.000
	Po100	1.500
	PoR502aCor	3.000
	Va200	3.000
Silk 7	*Bo50P20*	4.800
	Bo50P21	4.800
Silk 8	*POONGE125ML*	1.500

Tabela A.15: Emplastic_1209: Tempo de *setup* de tarefas do setor de serigrafia ($Setup_{j_i,j_k}$) [h]

	Tarefa Sucessora																			
Tarefa Predecessora	*Bo250P20*	*Bo250P21*	*Bo50P20*	*Bo50P21*	*CaEx*	*CaFu*	*CPRaLA3,5*	*CPRaLA3,52aCor*	*CPRaLA*	*MO100*	*MO250*	*Po100*	*Po250*	*PoR100*	*PoR250*	*PoR50*	*PoR502aCor*	*Tr*	*Va200*	*Va500*
Bo250P20			8	8	8	8	8	8	8	8		8				8			8	8
Bo250P21			8	8	8	8	8	8	8	8		8				8			8	8
Bo50P20	8	8					8	8	8	8	8	8	8						8	
Bo50P21	8	8					8	8	8	8	8	8	8						8	
CaEx	8	8											8						8	
CaFu	8	8											8						8	
CPRaLA3,5	8	8	8	8						8	8	8				8			8	
CPRaLA3,52aCor	8	8	8	8						8	8	8				8	8		8	
CPRaLA	8	8	8	8						8	8	8				8			8	
MO100	10	10	10	10			10	10	10		8	2				10			10	
MO250	2	2	10	10			10	10	10	8		10				10	10		10	
Po100	8	8	8	8			8	8	8		8		8		8	8	8	8	8	
Po250			8	8	8	8						8		8				8	8	8
PoR100													8		8			8		

Continua...

	Tarefa Sucessora																			
Tarefa Predecessora	*Bo250P20*	*Bo250P21*	*Bo50P20*	*Bo50P21*	*CaEx*	*CaFu*	*CPRaLA3,5*	*CPRaLA3,52aCor*	*CPRaLA*	*MO100*	*MO250*	*Po100*	*Po250*	*PoR100*	*PoR250*	*PoR50*	*PoR502aCor*	*Tr*	*Va200*	*Va500*
PoR250												8		8				8		
PoR50	8	8					8	8	8	8	8	8							8	
PoR502aCor								8			8	8							8	
Tr												8	8	8	8					
Va200	8	8	8	8	8	8	8	8	8	8	8	8	8			8	8			8
Va500	8	8											8						8	

Tabela A.16: Emplastic_1209: Tempo de *setup* de tarefas do setor de sopro ($Setup_{j_i,j_k}$) [h]

	Tarefa Sucessora																			
Tarefa Predecessora	*Fr-Bo250P21*	*Fr-Bo50P20*	*Fr-Bo50P21*	*Fr-CaEx*	*Fr-CaFu*	*Fr-PoR100*	*Fr-Tr*	*I100BPPH*	*I100NPEAD*	I200NPEAD	I250BPPH	I250NPEAD	I250NPEADPoR	I250NPPHCP	I500NPPH	I50NPEAD	I50NPPHCP	P500MPEAD	P500UPEAD	*PO250NPEAD*
FrBo250P21											2	2	2							
FrBo50P20																2				
FrBo50P21																2				
FrCaEx							10											10	10	10
FrCaFu							10											10	10	10
FrPoR100								2												
FrTr				8	8					8					8			8	8	8
I100BPPH						2			2											
I100NPEAD								2												
I200NPEAD							10													
I250BPPH	2											2	2	2						
I250NPEAD	2										2			2						
I250NPEADPoR	2										2			2						
I250NPPHCP											2	2	2							
I500NPPH							10											10	10	10

Continua...

	Tarefa Sucessora																			
Tarefa Predecessora	*FrBo250P21*	*FrBo50P20*	*FrBo50P21*	*FrCaEx*	*FrCaFu*	*FrPoR100*	*FrTr*	*I100BPPH*	*I100NPEAD*	I200NPEAD	I250BPPH	I250NPEAD	I250NPEADPoR	I250NPPHCP	I500NPPH	I50NPEAD	I50NPPHCP	P500MPEAD	P500UPEAD	*PO250NPEAD*
I50NPEAD		2	2														2			
I50NPPHCP																2				
P500MPEAD				10	10		10								10					10
P500UPEAD				10	10		10								10					10
PO250NPEAD				8	8		8								8			8	8	

Tabela A.17: Emplastic_1209: Demanda (Dem_j) [unidades]

Tarefa	Demanda
AB500	6.000
Bo250P20	36.000
Bo250P21	36.000
Bo50P20	120.000
Bo50P21	120.000
CaEx	4.000
CaFu	4.500
CPRaLA	8.000
CPRaLA3,52aCor	6.000
MO100	37.000
MO250	26.200
Po100	50.000
Po250	20.000
POONGE125ML	5.000
PoR100	20.100
PoR250	17.800
PoR50	40.000
PoR502aCor	40.000
Tr	10.500
Va200	36.000
Va500	17.000

Tabela A.18: Emplastic_1209: Ligação de tarefas ($LigTarefa_{j_i,j_k}$)

Tarefa Predecessora	Tarefa Sucessora
CPRaLA3,5	*CPRaLA3,52aCor*
FrBo250P21	*Bo250P21*
FrBo50P20	*Bo50P20*
FrBo50P21	*Bo50P21*
FrCaEx	*CaEx*
FrCaFu	*CaFu*
FrPoR100	*PoR100*
FrTr	*Tr*
I100BPPH	*MO100*
I100NPEAD	*Po100*
I200NPEAD	*Va200*
I250BPPH	*MO250*
I250NPEAD	*Po250*
I250NPEADPoR	*PoR250*
I250NPPHCP	*Bo250P20*
I500NPPH	*Va500*
I50NPEAD	*PoR50*
I50NPPHCP	*FrBo250P21*
P500MPEAD	*CPRaLA*
P500UPEAD	*CPRaLA3,5*
PO250NPEAD	*POONGE125ML*
PoR50	*PoR502aCor*

Tabela A.19: Emplastic_1209: Data entrega (DE_j)

Tarefa	Turno
AB500	*23*
Bo250P20	*17*
Bo250P21	*21*
Bo50P20	*17*
Bo50P21	*21*
CaEx	*21*

Continua...

Tarefa	Turno
CaFu	*24*
CPRaLA	*4*
CPRaLA3,52aCor	*21*
MO100	*4*
MO250	*21*
Po100	*9*
Po250	*5*
POONGE125ML	*6*
PoR100	*21*
PoR250	*21*
PoR50	*16*
PoR502aCor	*16*
Tr	*23*
Va200	*21*
Va500	*21*

Máximo funcionários ($FDisp_{t,s}$): $\begin{cases} 'Sopro' : & 5 \\ 'Serigrafia' : & 15 \end{cases}, \forall t$

Tabela A.20: Emplastic_1209: Funcionários para qualquer tarefa nas máquinas ($F_{j,m}$)

Máquina	Funcionários
BPS	1
PZ	1
RG10	1
RG4	1*
RG5	1
RG6	1
Silk 1	3
Silk 2	3
Silk 3	3
Silk 5	3

Continua...

* O par P500MPEAD, RG4 exige 2 funcionários

Máquina	Funcionários
Silk 6	3
Silk 7	3
Silk 8	*3*

* O par P500MPEAD, RG4 exige 2 funcionários

Tabela A.21: Emplastic_1209: Moldes disponíveis (NM_j)

Tarefa	Moldes
AB500	1
Bo250P20	3
Bo250P21	3
Bo50P20	4
Bo50P21	4
CaEx	1
CaFu	1
CPRaLA	1
CPRaLA3,5	1
CPRaLA3,52aCor	2
FrBo250P21	1
FrBo50P20	1
FrBo50P21	1
FrCaEx	1
FrCaFu	1
FrPoR100	1
FrTr	2
I100BPPH	1
I100NPEAD	1
I200NPEAD	1
I250BPPH	1
I250NPEAD	1
I250NPEADPoR	1
I250NPPHCP	1
I500NPPH	1
I50NPEAD	1

Continua...

Tarefa	Moldes
I50NPPHCP	1
MO100	1
MO250	2
P500MPEAD	1
P500UPEAD	1
Po100	3
Po250	3
PO250NPEAD	1
POONGE125ML	1
PoR100	1
PoR250	1
PoR50	1
PoR502aCor	1
Tr	1
Va200	3
Va500	1

Tabela A.22: Emplastic_1209: Tarefa usa material ($JMat_{j,r}$)

	Material		
Tarefa	*PPHCP*	*PEAD*	*PPH*
AB500			x
FrBo250P21	x		
FrBo50P20	x		
FrBo50P21	x		
FrCaEx			x
FrCaFu			x
FrPoR100		x	
I100BPPH			x
I100NPEAD		x	
I200NPEAD		x	
I250BPPH			x
I250NPEAD		x	
I250NPEADPoR		x	

Continua...

	Material		
Tarefa	*PPHCP*	*PEAD*	*PPH*
I250NPPHCP	x		
I500NPPH			x
I50NPEAD		x	
I50NPPHCP	x		
MO100			x
MO250			x
P500MPEAD		x	
P500UPEAD		x	

Máximo materiais ($Matmax$): 4

Início mínimo tarefa ($JMin_j$): Todas as tarefas podem iniciar no primeiro turno.

Início mínimo máquina ($MMin_m$): Todas as máquinas podem começar no primeiro turno.

Tabela A.23: Emplastic_1209: Setor de tarefa ($JSetor_{j,s}$)

	Setor	
Tarefa	*Serigrafia*	*Sopro*
AB500		x
Bo250P20	x	
Bo250P21	x	
Bo50P20	x	
Bo50P21	x	
CaEx	x	
CaFu	x	
CPRaLA	x	
CPRaLA3,5	x	
CPRaLA3,52aCor	x	
FrBo250P21		x
FrBo50P20		x
FrBo50P21		x

Continua...

	Setor	
Tarefa	*Serigrafia*	*Sopro*
FrCaEx		x
FrCaFu		x
FrPoR100		x
FrTr		x
I100BPPH		x
I100NPEAD		x
I200NPEAD		x
I250BPPH		x
I250NPEAD		x
I250NPEADPoR		x
I250NPPHCP		x
I500NPPH		x
I50NPEAD		x
I50NPPHCP		x
MO100	x	
MO250	x	
P500MPEAD		x
P500UPEAD		x
Po100	x	
Po250	x	
PO250NPEAD		x
POONGE125ML	x	
PoR100	x	
PoR250	x	
PoR50	x	
PoR502aCor	x	
Tr	x	
Va200	x	
Va500	x	

Tabela A.24: Emplastic_1209: Horas normais e extras disponíveis por semana na serigrafia ($HTurno_{t,s}$ e $HExt_{t,s}$) [h]

	Segunda	*Terça*	*Quarta*	*Quinta*	*Sexta*	*Sábado*	*Domingo*
Normal	8	8	8	8	8	0	0
Extra	2	2	2	2	2	8	0

Tabela A.25: Emplastic_1209: Horas normais e extras disponíveis por semana no setor sopro ($HTurno_{t,s}$ e $HExt_{t,s}$) [h]

	1o Turno (Seg)	*2o Turno (Seg)*	*3o Turno (Seg)*	*1o Turno (Ter)*	*2o Turno (Ter)*	*3o Turno (Ter)*	*1o Turno (Qua)*	*2o Turno (Qua)*	*3o Turno (Qua)*	*1o Turno (Qui)*	*2o Turno (Qui)*	*3o Turno (Qui)*	*1o Turno (Sex)*	*2o Turno (Sex)*	*3o Turno (Sex)*	*1o Turno (Sab)*	*2o Turno (Sab)*	*3o Turno (Sab)*	*Domingo*
Normal	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	0	0
Extra	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Dia do período inicial: Domingo

Estoque inicial (Est_j): Não há estoque inicial disponível

A.3 Emplastic_0210

Tarefas (J): AF250P06, AF50P06, BO250P03, BO250P04, BO250P05, BO50P03, BO50P04, BO50P05, FRBO250P04, FRBO250P05, FRBO50P04, FRBO50P05, FRGL100, FRPO250P02, FRPO250P03, FRPOBI, FRPOBI1, FRPOGE125M, FRPR100P01, FrTr, GL100, PO100P01, PO250P02, PO250P03, POBI, POBI1, POGE125ML, POR100P01, Tr

Máquinas (M): BPS, INJ1, INJ2, PZ, RG1, RG10, RG3, RG4, RG5, RG6, RG7, 'Silk 1', 'Silk 2', 'Silk 3', 'Silk 5', 'Silk 6', 'Silk 7', 'Silk 8'

Períodos (P): 0..26

Turnos (T): 1..81

Setores (S): Sopro, Serigrafia

Materiais (R): PPHCP, PEAD, PEBD, PPH, BPEAD

Tabela A.26: Emplastic_0210: Produtividade ($Prod_{j,m}$) [unidades/h]

Máquina	Tarefa	Produtividade
BPS	*FrTr*	772
RG10	*FRBO250P04*	982
	FRBO250P05	982
	FRPO250P02	982
	FRPO250P03	982
RG4	*FRPOGE125ML*	240
	FrTr	772
RG5	*FRBO50P04*	1.800
	FRBO50P05	1.800
RG6	*FRGL100*	1.108
	FRPR100P01	1.440
RG7	*FRPOBI*	240
	FRPOBI1	240
Silk 1	*AF250P06*	3.000
	AF50P06	3.000
	BO250P03	3.000
	BO250P04	3.000
	BO250P05	3.000
	BO50P03	3.000
	BO50P04	3.000
	BO50P05	3.000
	PO250P02	1.500
	PO250P03	1.500
Silk 2	*AF250P06*	3.000
	BO250P03	3.000
	BO250P04	3.000
	BO250P05	3.000

Continua...

Máquina	Tarefa	Produtividade
	PO250P02	1.500
	PO250P03	1.500
Silk 3	*GL100*	2.400
	PO100P01	2.400
	PO250P02	2.400
	PO250P03	2.400
	POR100P01	2.400
	Tr	2.100
Silk 5	*AF250P06*	3.000
	AF50P06	3.000
	BO250P03	3.000
	BO250P04	3.000
	BO250P05	3.000
	BO50P03	3.000
	BO50P04	3.000
	BO50P05	3.000
	GL100	3.000
	PO100P01	1.500
Silk 6	*AF250P06*	3.000
	AF50P06	3.000
	BO50P03	3.000
	BO50P04	3.000
	BO50P05	3.000
	GL100	3.000
	PO100P01	1.500
Silk 7	*AF50P06*	4.800
	BO50P03	4.800
	BO50P04	4.800
	BO50P05	4.800
Silk 8	*POBI*	1.500
	POBI1	1.500
	POGE125ML	1.500

Tabela A.27: Emplastic_0210: Tempo de *setup* de tarefas do setor de serigrafia ($Setup_{j_i,j_k}$) [unidades/h]

	Tarefa Sucessora																
Tarefa Predecessora	*Tr*	*PO100P01*	*POBI*	*PO250P03*	*PO250P02*	*POBI1*	*POGE125ML*	*AF50P06*	*AF250P06*	*BO50P03*	*BO250P03*	*POR100P01*	*BO50P04*	*BO250P04*	*BO250P05*	*BO50P05*	*GL100*
Tr		4		4	4							4					4
PO100P01	4			4	4			4	4	4	4		4	4	4	4	
POBI							4										
PO250P03	4	4						4		4		4	4			4	4
PO250P02	4	4						4		4		4	4			4	4
POBI1							4										
POGE125ML			4			4											
AF50P06		4		4	4				4		4			4	4		4
AF250P06		4						4		4			4			4	4
BO50P03		4		4	4				4		4			4	4		4
BO250P03		4						4		4			4			4	4
POR100P01	4			4	4												
BO50P04		4		4	4				4		4			4	4		4
BO250P04		4						4		4			4			4	4
BO250P05		4						4		4			4			4	4
BO50P05		4		4	4				4		4			4	4		4
GL100	4			4	4			4	4	4	4		4	4	4	4	

Tabela A.28: Emplastic_0210: Tempo de *setup* de tarefas do setor de sopro ($Setup_{j_i,j_k}$) [h]

	Tarefa Sucessora						
Tarefa Predecessora	*FrTr*	*FRBO250P04*	*FRPO250P03*	*FRPO250P02*	*FRBO250P05*	*FRGL100*	*FRPOGE125ML*
FrTr							8

Continua...

	Tarefa Sucessora						
Tarefa Predecessora	*FrTr*	*FRBO250P04*	*FRPO250P03*	*FRPO250P02*	*FRBO250P05*	*FRGL100*	*FRPOGE125ML*
FRPR100P01						1	
FRBO250P04			1	1			
FRPO250P03		1			1		
FRPO250P02		1			1		
FRBO250P05			1	1			
FRPOGE125ML	8						

Tabela A.29: Emplastic_0210: Demanda (Dem_j) [unidades]

Tarefa	Demanda
AF250P06	85.000
AF50P06	45.500
BO250P03	36.000
BO250P04	36.000
BO250P05	36.000
BO50P03	120.000
BO50P04	120.000
BO50P05	120.000
GL100	10.000
PO100P01	50.000
PO250P02	20.000
PO250P03	20.000
POBI	25.000
POBI1	25.000
POGE125ML	5.000
POR100P01	40.200
Tr	21.500

Tabela A.30: Emplastic_0210: Ligação de tarefas ($LigTarefa_{j_i,j_k}$)

Tarefa Predecessora	Tarefa Sucessora
FRBO250P04	*BO250P04*
FRBO250P05	*BO250P05*
FRBO50P04	*BO50P04*
FRBO50P05	*BO50P05*
FRGL100	*GL100*
FRPO250P02	*PO250P02*
FRPO250P03	*PO250P03*
FRPOBI	*POBI*
FRPOBI1	*POBI1*
FRPOGE125ML	*POGE125ML*
FRPR100P01	*POR100P01*

Tabela A.31: Emplastic_0210: Data entrega (DE_j)

Tarefa	Turno
AF250P06	*8*
AF50P06	*8*
BO250P03	*8*
BO250P04	*27*
BO250P05	*22*
BO50P03	*8*
BO50P04	*27*
BO50P05	*22*
GL100	*27*
PO100P01	*6*
PO250P02	*27*
PO250P03	*27*
POBI	*27*
POBI1	*26*
POGE125ML	*27*
POR100P01	*27*
Tr	*19*

Máximo funcionários ($FDisp_{t,s}$): $\begin{cases} 'Sopro': & 5 \\ 'Serigrafia': & 15 \end{cases}, \forall t$

Tabela A.32: Emplastic_0210: Funcionários para qualquer tarefa nas máquinas de Sopro ($F_{j,m}$)

Máquina	Funcionários
BPS	1
PZ	1
RG10	1
RG4	1
RG5	1
RG6	1

Tabela A.33: Emplastic_0210: Funcionários para as tarefas nas máquinas de Serigrafia ($F_{j,m}$)

	Máquina						
Tarefa	*Silk 1*	*Silk 2*	*Silk 3*	*Silk 5*	*Silk 6*	*Silk 7*	*Silk 8*
AF250P06	4	4	4	4	4		
AF50P06	4			4	4	4	
BO250P03	3	3	3	3	3		
BO250P04	3	3	3	3	3		
BO250P05	3	3	3	3	3		
BO50P03	3			3	3	3	
BO50P04	3			3	3	3	
BO50P05	3			3	3	3	
PO100P01			3	3	3		
PO250P02	3	3	3	3	3		
PO250P03	3	3	3	3	3		
POBI							2
POBI1							2
POGE125ML							3

Continua...

	Máquina						
Tarefa	*Silk 1*	*Silk 2*	*Silk 3*	*Silk 5*	*Silk 6*	*Silk 7*	*Silk 8*
POR100P01			3	3	3		
Tr			4				

Tabela A.34: Emplastic_0210: Moldes disponíveis (NM_j)

Tarefa	Moldes
AF250P06	4
AF50P06	4
BO250P03	3
BO250P04	3
BO250P05	3
BO50P03	4
BO50P04	4
BO50P05	4
FRBO250P04	1
FRBO250P05	1
FRBO50P04	1
FRBO50P05	1
FRGL100	1
FRPO250P02	1
FRPO250P03	1
FRPOBI	1
FRPOBI1	1
FRPOGE125ML	1
FRPR100P01	1
FrTr	2
GL100	3
PO100P01	3
PO250P02	3
PO250P03	3
POBI	1

Continua...

Tarefa	Moldes
POBI1	1
POGE125ML	1
POR100P01	1
Tr	1

Tabela A.35: Emplastic_0210: Tarefa usa material ($JMat_{j,r}$)

	Material	
Tarefa	*PPHCP*	*PEAD*
FRBO250P04	x	
FRBO250P05	x	
FRBO50P04	x	
FRBO50P05	x	
FRPO250P02		x
FRPO250P03		x
FRPR100P01		x

Máximo materiais ($Matmax$): 4

Início mínimo tarefa ($JMin_j$): Todas as tarefas podem iniciar no primeiro turno.

Início mínimo máquina ($MMin_m$): Todas as máquinas podem começar no primeiro turno.

Tabela A.36: Emplastic_0210: Setor de tarefa ($JSetor_{j,s}$)

	Setor	
Tarefa	*Serigrafia*	*Sopro*
AF250P06	x	
AF50P06	x	
BO250P03	x	
BO250P04	x	
BO250P05	x	

Continua...

	Setor	
Tarefa	*Serigrafia*	*Sopro*
BO50P03	x	
BO50P04	x	
BO50P05	x	
FRBO250P04		x
FRBO250P05		x
FRBO50P04		x
FRBO50P05		x
FRGL100		x
FRPO250P02		x
FRPO250P03		x
FRPOBI		x
FRPOBI1		x
FRPOGE125ML		x
FRPR100P01		x
FrTr		x
GL100	x	
PO100P01	x	
PO250P02	x	
PO250P03	x	
POBI	x	
POBI1	x	
POGE125ML	x	
POR100P01	x	
Tr	x	

Tabela A.37: Emplastic_1209: Horas normais e extras disponíveis por semana na serigrafia ($HTurno_{t,s}$ e $HExt_{t,s}$)

	Segunda	*Terça*	*Quarta*	*Quinta*	*Sexta*	*Sábado*	*Domingo*
Normal	8	8	8	8	8	0	0

Continua...

	Segunda	*Terça*	*Quarta*	*Quinta*	*Sexta*	*Sábado*	*Domingo*
Extra	2	2	2	2	2	8	8

Tabela A.38: Emplastic_1209: Horas normais e extras disponíveis por semana no setor sopro ($HTurno_{t,s}$ e $HExt_{t,s}$)

	1o Turno (Seg)	*2o Turno (Seg)*	*3o Turno (Seg)*	*1o Turno (Ter)*	*2o Turno (Ter)*	*3o Turno (Ter)*	*1o Turno (Qua)*	*2o Turno (Qua)*	*3o Turno (Qua)*	*1o Turno (Qui)*	*2o Turno (Qui)*	*3o Turno (Qui)*	*1o Turno (Sex)*	*2o Turno (Sex)*	*3o Turno (Sex)*	*1o Turno (Sab)*	*2o Turno (Sab)*	*3o Turno (Sab)*	*1o Turno (Dom)*	*2o Turno (Dom)*	*3o Turno (Dom)*
Normal	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	0	0	*0*	*0*
Extra	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8	*8*	*8*

Dia do período inicial: Segunda-feira

Estoque inicial (Est_j): Não há estoque inicial disponível