• U C •

FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

DEPARTAMENTO DE
ENGENHARIA MECÂNICA

# Lot Sizing and Scheduling

# A case study in the plastic Enjection Industry

Dissertação apresentada para a obtenção do grau de Mestre em Engenharia e Gestão Industrial

**Autor**

**João Marcelo Figueira Veríssimo**

**Orientador**

**Professor Doutor Cristóvão Silva**

| | |
|---|---|
| **Júri** | |
| **Presidente** | **Professor Doutor Pedro Mariano Simões Neto**<br>**Professor Auxiliar da Universidade de Coimbra** |
| **Vogais** | **Professor Doutor Cristóvão Silva**<br>**Professor Auxiliar da Universidade de Coimbra**<br>**Professor Doutor Luís Miguel Domingues Fernandes Ferreira**<br>**Professor Auxiliar da Universidade de Coimbra** |
| **Orientador** | **Professor Doutor Cristóvão Silva**<br>**Professor Auxiliar da Universidade de Coimbra** |

**Coimbra, Setembro, 2016**

# AGRADECIMENTOS

O trabalho só foi possível graças à colaboração e apoio de algumas pessoas, às quais não posso deixar de prestar o meu reconhecimento.

Ao meu orientador, Professor Doutor Cristóvão Silva e ao Professor Doutor Carlos M. Fonseca pelo contributo técnico e científico para que este trabalho possa ter sido realizado.

A todos os meus amigos, que de alguma forma contribuíram para que este trabalho fosse realizado.

# RESUMO

Atualmente verifica-se que a indústria está em constantes mudanças, sendo cada vez mais inteligente e flexível. Nos processos industriais, a tomada de decisão é fulcral e pode ser facilitada com recurso a diversos métodos e técnicas computacionais. A otimização dos processos suscita um interesse elevado no tecido industrial, uma vez que permite aumentar a competitividade e o lucro.

Assim sendo, a principal motivação deste trabalho passa pelo desenvolvimento de uma ferramenta que possa auxiliar a tomada de decisão no tecido industrial, visando a otimização dos processos, racionalização de recursos e, consequentemente ganhos em eficiência no processo produtivo.

A ferramenta foi obtida através da construção de um modelo matemático de Programação Linear Inteira Mista e tem como objetivo minimizar o atraso total das tarefas programadas, e consequentemente, encontrar a melhor sequência de agendamento da produção num ambiente de *n* tarefas em *m* máquinas paralelas idênticas. Para além disto, o sequenciamento é obtido tendo em conta o loteamento de tarefas, nomeadamente o binómio molde-máquina. Devido à elevada natureza combinatória estes problemas são considerados como NP-Hard.

O modelo proposto foi implementado na linguagem de modelagem matemática AMPL (Algebraic Mathematical Programming Language), recorrendo posteriormente ao software GLPK (GNU Linear Programming Kit).

Ao longo do trabalho foi possível testar duas instâncias, sendo a primeira com 32 tarefas e 2 máquinas e a segunda com 10 tarefas e 2 máquinas. A ferramenta é eficaz na tomada de decisão para instâncias de pequena dimensão, no entanto, quando se aumenta a quantidade de informação (tarefas e máquinas), o problema não é resolvido num tempo computacional útil.

**Palavras-chave:** Programação Linear Inteira Mista, Loteamento, Sequenciamento, Tarefas, Máquinas Paralelas Idênticas, AMPL, GLPK, NP-Hard.

# ABSTRACT

Currently all the industry sectors are constantly changing and becoming intelligent and flexible. In industrial processes decision-making is critical and can be improved significantly using various methods and computational techniques. The optimization has huge advantages in the industrial environment, as it allows an increase in competitiveness and profit.

Therefore, the main motivation of this work involves the development of a tool that can assist decision-making processes in the industry, with prime focus on process optimization and resource rationalization in order to get efficiency in the overall production.

The tool is obtained by building a mathematical model of mixed linear programming integer that aims to minimize the total tardiness of scheduled jobs, and thus finding the best scheduling sequence of production giving *n* jobs over *m* identical parallel machines. In addition, the sequence is achieved regarding the subdivision of jobs, including binomial mold-machine. Because of the high combinatorial nature of these problems they are considered to be NP-Hard.

The proposed model was implemented in mathematical modeling language AMPL (Algebraic Mathematical Programming Language), then using the GLPK software (GNU Linear Programming Kit).

Throughout the work, it was possible to test two instances, the first one with 32 jobs and 2 machines and the second 10 jobs and 2 machines. The tool is effective in decision making for small instances, however, when increasing the amount of information (jobs and machines), the problem is not solved in a useful computational time.

**Keywords** Mixed Integer Programming, Jobs, Identical Parallel Machines, AMPL, GLPK, NP-Hard.

# Índice

# ÍNDICE DE FIGURAS

# ÍNDICE DE TABELAS

# SIMBOLOGIA

# SIGLAS

AMPL – A Mathematical Programming Language

GLPK – GNU Linear Programming Kit

JIT – Just-in-Time

JS – Job Shop

JSS – Job Shop Scheduling

NP-Hard – Nondeterministic Polynomial Time

PL – Programação Linear

PLB – Programação Linear Binária

PLI – Programação Linear Inteira

PLIM – Programação Linear Inteira Mista

PLIP – Programação Linear Inteira Pura

PME – Pequena e Média Dimensão

PCP – Planeamento e Controlo da Produção

PMP – Plano Mestre de Produção

# 1. INTRODUÇÃO

A elevada competitividade que se verifica nos sistemas produtivos tem vindo a aumentar de forma exponencial ao longo dos últimos anos. A indústria tem vindo a sofrer constantes mudanças ao longo do tempo, levando à necessidade de evoluir e adaptar-se às exigências do mercado e, por isso, é necessário reconsiderar a estratégia organizacional (nomeadamente a nível de processos produtivos) para conseguir enfrentar as ameaças externas, bem como satisfazer as exigências dos clientes. [1]

A crescente competitividade que se verifica nos dias de hoje faz com que a indústria seja cada vez mais inteligente e flexível, assim sendo, é necessário adotar medidas e estratégias que possibilitem às empresas criar diferenciação no mercado. [2]

Devido aos avanços tecnológicos no tecido industrial, as empresas encontram-se cada vez mais num meio extremamente competitivo. O principal foco é proporcionar aos clientes serviços de qualidade satisfazendo as suas necessidades e, para isso, é determinante para um bom desempenho de um sistema produtivo, existir um eficaz Planeamento e Controlo da Produção (PCP). O sucesso e eficácia do PCP tem como ferramenta base o Plano Mestre de Produção (PMP). Este define o que deverá ser fabricado, tendo em conta as necessidades dos clientes e as capacidades da empresa, ou seja, é um plano detalhado de produção para produtos individuais em que geralmente o horizonte de planeamento é dividido em períodos semanais. [3]

O planeamento e a programação de tarefas em meio laboral distinguem-se na medida em que o planeamento define o que deve ser feito tendo em conta as restrições, e a programação especifica como e quando deverá ser feito. [4]

Por vezes existem sistemas de produção que têm de respeitar prioridades e obedecer a determinadas restrições (muitas vezes impostas pelos clientes), sendo necessário um método eficiente para obter resultados positivos no planeamento e programação das tarefas.

O problema de loteamento e sequenciamento é designado por "lot size scheduling problem". Técnicas de loteamento podem ser aplicadas de modo a melhorar a utilização dos recursos de produção, aumentando a eficiência no processo produtivo.

Assim, surge a necessidade do sequenciamento que consiste em definir prioridades ou ordenar a execução de um conjunto de tarefas, de modo a atingir um determinado objetivo, satisfazendo as restrições existentes. [5]

A ordem pela qual as tarefas são executadas é extremamente importante tendo em conta os recursos disponíveis na empresa, assim, surge a necessidade de uma calendarização para um determinado período temporal. [4]

Por vezes associado a problemas de agendamento da produção surge o conceito Just-in-Time (JIT). O JIT é um método de gestão da produção com foco na redução de stocks (em qualquer ponto do processo produtivo), mas também em alguns elementos determinantes na melhoria do desempenho: a melhoria continua, o controlo de qualidade e eliminação de desperdícios. A filosofia JIT segue o princípio de que produzimos apenas o que é necessário, quando for necessário e na quantidade necessária. [6]

O tema deste trabalho centra-se no problema de loteamento e sequenciamento de operações em ambiente Job Shop (JS), onde o principal objetivo é desenvolver um modelo matemático que possa ser aplicado em diversos cenários a determinados níveis industriais, para seja possível gerir de forma eficiente os recursos disponíveis, proporcionando uma tomada de decisão eficaz, e consequentemente uma redução de custos, aumento de produtividade e diminuição do tempo para conclusão das tarefas em máquinas paralelas idênticas (os tempos de processamento de cada tarefa são iguais para todas as máquinas). Problemas deste tipo são considerados NP-Hard (Nondeterministic polynomial time), devido à sua complexidade e esforço computacional, sendo necessário aplicar técnicas de otimização para obter uma solução ótima. [7]

Na abordagem ao problema de otimização foi usado um modelo matemático de programação linear inteira mista (PLIM) desenvolvido no editor de texto GUSEK, e implementado na linguagem de modelagem matemática AMPL (Algebraic Mathematical Programming Language), recorrendo posteriormente ao software GNU Linear Programming Kit (GLPK). O modelo matemático implementado é baseado na proposta de Mohamed. K. Omar, Siew C. Teo e Yasothei Suppiah, que tem como objetivo minimizar o atraso total das tarefas programadas, e consequentemente encontrar a melhor sequência de agendamento da produção num ambiente JS, em diferentes linhas de produção. [8]

Esta ferramenta de apoio à decisão permite às empresas aumentar a sua flexibilidade na medida em que conseguirão planear e otimizar de forma eficaz as suas

tarefas, cumprindo atempadamente os prazos impostos pelos clientes, garantindo estabilidade no PMP.

Relativamente à estrutura da dissertação, podemos encontrar no capítulo 1 a introdução do trabalho, no capitulo 2 uma breve revisão dos conceitos mais importantes que estão inerentes ao desenvolvimento do mesmo. No capítulo 3, aborda-se essencialmente a descrição do problema e o processo de construção do modelo matemático usado. No capítulo 4, podemos encontrar os resultados obtidos, bem como a discussão dos mesmos e, por último, no capítulo 5, as conclusões obtidas ao longo do desenvolvimento do trabalho.

# 2. ENQUADRAMENTO TEÓRICO

## 2.1. Indústria de moldes para matérias plásticas

A indústria de moldes para matérias plásticas em Portugal surgiu em 1943, na Marinha Grande, numa pequena empresa de moldes para vidro, por iniciativa de Aníbal H. Abrantes. Dois anos mais tarde, Abrantes produziu o primeiro molde de injeção para plástico. Com o passar dos anos começaram a surgir novas empresas produtoras de moldes para plásticos nomeadamente na Marinha Grande e Oliveira de Azeméis. Devido à importação da tecnologia, em 1955, começaram-se a dar os primeiros passos na exportação, nomeadamente com a venda dos primeiros moldes à Grã-Bretanha. Em 1980, a indústria de moldes para matérias plásticas já exportava para mais de 50 países e, somente na Marinha Grande, existiam 54 empresas com um total de 2000 postos de trabalho. [9]

Devido à elevada procura externa, aliado a um conjunto de competências e capacidade produtiva que a Indústria Portuguesa de Moldes oferece aos seus clientes, esta tem vindo a crescer e a consolidar a sua posição no mercado internacional. Atualmente o sector Português de Moldes detém 450 empresas de pequena e média dimensão (PME), dedicadas à conceção, desenvolvimento e fabrico de moldes e ferramentas especiais, empregando 8000 trabalhadores, localizadas essencialmente nas regiões da Marinha Grande e Oliveira de Azeméis. Portugal atualmente exporta mais de 85% da sua produção total, sendo o 8º a nível global, 3º a nível europeu, na área de fabricação de moldes para injeção de plástico. [9]

Conforme a figura 1, verificamos que em 2015 a exportação atingiu um valor superior a 590 milhões de euros (o melhor ano de sempre da indústria em termos de produção e exportação), demonstrando a elevada capacidade de adaptação às exigências dos clientes, das tecnologias e dos mercados. Verifica-se ainda que existe uma tendência de crescimento, sendo que o saldo da balança comercial passou de cerca 230 milhões de euros em 2005 para cerca de 440 milhões de euros em 2015.

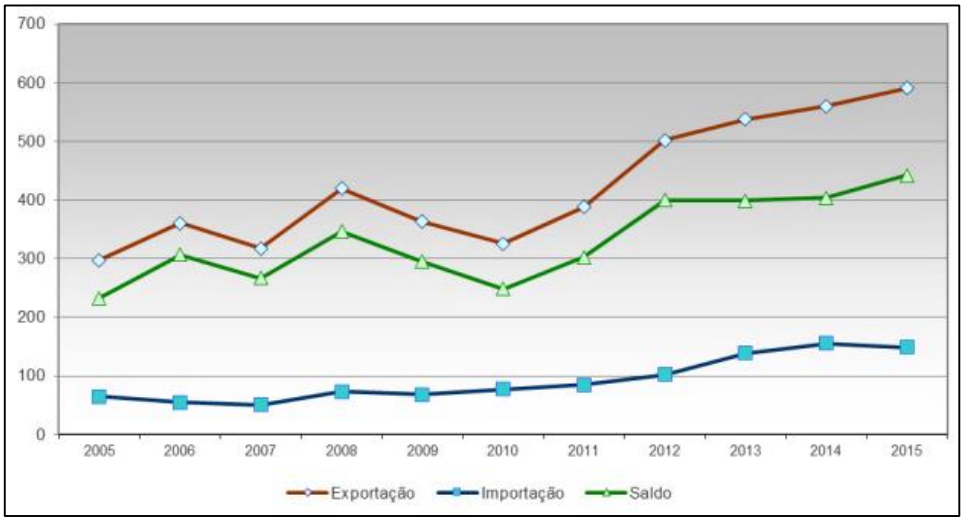


**Figura 1-Balança Comercial da Indústria Portuguesa de Moldes. (CEFAMOL)**

Relativamente à figura 2 podemos observar uma forte tendência para a exportação, nunca inferior a 75%. Nos últimos dez anos verifica-se que existe um crescimento acentuado das exportações. No entanto, o decréscimo que se verifica de 2009 a 2011 é consequência da crise conjuntural que se viveu nesses anos. [9]

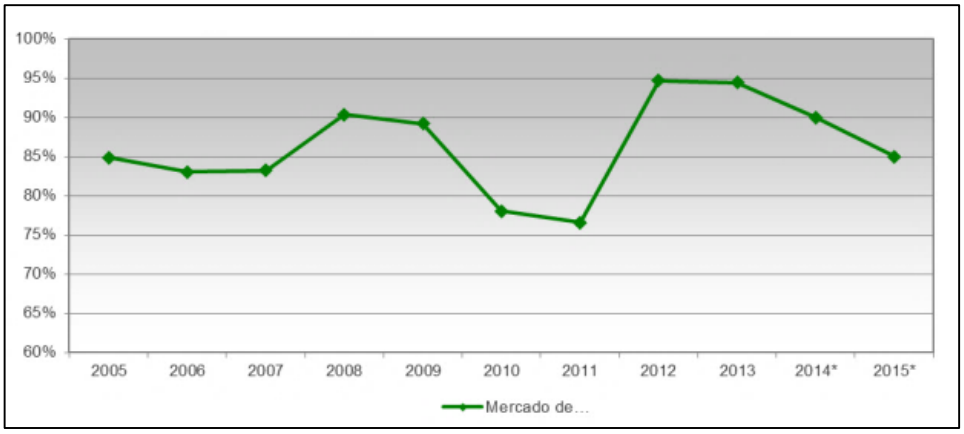


**Figura 2-Evolução do Mercado de Exportação da Indústria Portuguesa de Moldes. (CEFAMOL)**

Na figura 3, observamos ainda uma predominância do mercado europeu, representando nos últimos anos, em média, 79% do mercado total de exportações. Importa salientar que o decréscimo que ocorreu nas exportações para a América do Norte é consequência da deslocalização de empresas clientes para países onde o custo de mão de obra é mais barato, bem como pela forte depreciação do dólar americano face ao euro. [9]

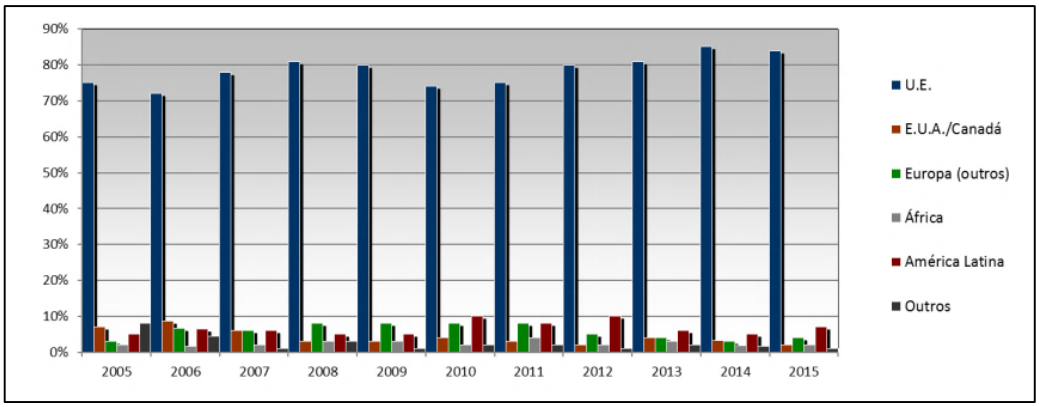


**Figura 3-Exportações por zona Económica da Indústria Portuguesa de Moldes. (CEFAMOL)**

Pela análise da figura 4, destaca-se ainda que os principais mercados das exportações portuguesas foram: Alemanha (22%), Espanha (19%), França (18%), Reino Unido (5%) e Republica Checa (5%).

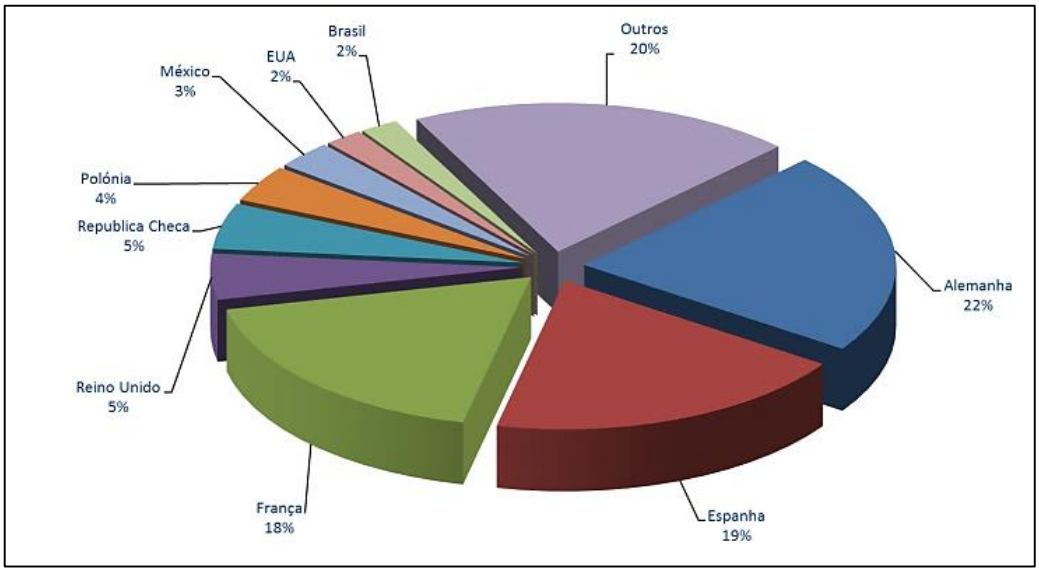


**Figura 4-Principais mercados de exportação da Indústria Portuguesa de Moldes. (CEFAMOL)**

Por fim, a figura 5, demonstra que a indústria automóvel tem sido fundamental para o desenvolvimento do sector, com um peso de 74%. Outra indústria que contribui para a evolução do sector é a indústria das embalagens, neste momento com 10% da produção nacional de moldes.

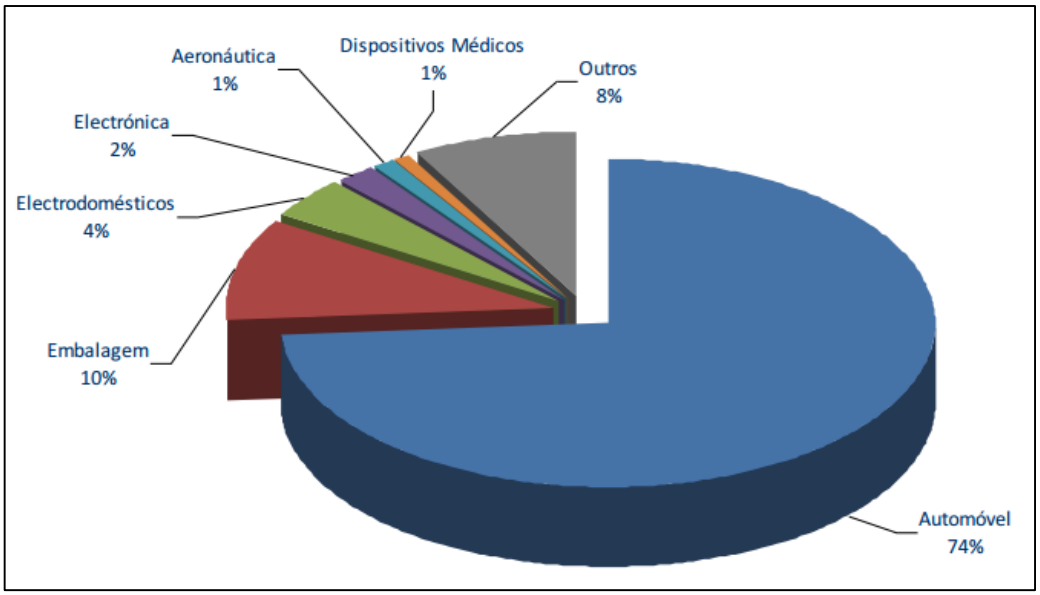


**Figura 5-Principais clientes da Indústria Portuguesa de Moldes. (CEFAMOL)**

## 2.2. Loteamento e Sequenciamento

O problema de loteamento e sequenciamento é designado por "lot size scheduling problem". Neste tipo de problemas as tarefas são agrupadas em lotes (famílias com as mesmas características), aos quais se encontra associada uma data de entrega ao cliente. Quando se pretende fabricar um produto referente a uma determinada família diferente da anterior é necessário modificar algumas configurações dos equipamentos. A necessidade de ajustar os equipamentos, por vezes, acarreta demasiados custos e consomem tempo. [10]

As técnicas de loteamento na programação da produção focam-se essencialmente na redução dos custos financeiros, bem como na diminuição do tempo necessário na produção. Esta redução pode ser obtida quando se agrupam tarefas do mesmo lote, não sendo preciso um tempo de preparação da máquina (tempo setup). O tempo de setup pode ser independente ou dependente da sequência da família das tarefas. Caso seja

independente, o tempo de setup é determinado pela próxima tarefa a ser processada, caso seja dependente, o setup é determinado pela tarefa que foi processada e pela tarefa seguinte. [11]

A eficácia do processamento de tarefas pode ser obtida quando consideramos processar os lotes com tarefas da mesma família porque será necessário um menor número de setups (diminuição da troca de moldes). No entanto, ao processarmos grandes lotes podemos estar a comprometer a data de entrega, devido ao atraso no processamento das tarefas. [11]

Consoante as características das operações a realizar para uma determinada tarefa, por vezes é mais barato processar as tarefas em lotes do que individualmente. É necessário que o chão de fábrica seja flexível para poder suportar elevadas cargas de trabalho e informação que por vezes chega em tempo real. [11]

Perante isto, é necessário otimizar os processos, nomeadamente ao nível da minimização do número de alterações e da melhor sequência de processamento sem que haja prejuízo nas datas de entrega.

Sequenciamento é um processo de atribuição de tarefas a uma determinada máquina ou centro de trabalho de modo a serem processadas num determinado período de tempo. É considerado um dos mais antigos problemas de otimização, tendo sido estudado ao longo de vários anos, originando diversificadas abordagens de resolução. Em ambientes de fabrico reais muitas destas abordagens tornam-se impraticáveis devido à dinâmica existente no chão de fábrica, tal como, restrições complexas ou perturbações inesperadas. A informação em tempo real é uma característica intrínseca no tecido industrial; assim, por vezes, é necessária uma reconsideração dos planos pré-estabelecidos, e consequentemente uma constante verificação da disponibilidade de recursos existentes. [12]

Em seguida observa-se a evolução do conceito de sequenciamento ("scheduling") ao longo do tempo. [13]

**Tabela 1-Definição Scheduling**

| Autor | Definição |
|---|---|
| **Baker (1974)** | "Scheduling é a atribuição de recursos no tempo para o processamento de um conjunto de tarefas" |

| | |
|---|---|
| **French (1982)** | "Scheduling consiste em encontrar uma sequência de passagem das tarefas pelos recursos, correspondendo a um plano exequível e ótimo em relação a um qualquer critério de otimização adotado" |
| **Artiba e Elmaghraby (1997)** | "Scheduling consiste na determinação do sequenciamento das tarefas ou ordens de fabrico no tempo e na sua atribuição aos respetivos recursos" |
| **Portmann (1997)** | "Scheduling pode ser descrito como a definição de tempos de início e de conclusão e a atribuição dos recursos a cada tarefa de um dado conjunto, obedecendo às várias restrições dos recursos e/ou tarefas" |
| **Blazewicz et al. (2001)** | "Os problemas de scheduling podem ser definidos duma forma geral como o problema de atribuição de recursos ao longo do tempo para a realização de um conjunto de tarefas" |
| **Pinedo (2008)** | "Scheduling é definido como a alocação de recursos a tarefas ao longo do tempo, e é um processo de tomada de decisão com o intuito de otimizar um ou mais objetivos" |

## 2.3. Problemas de Sequenciamento

### Conceitos

**Tempo de processamento**

- Tempo no qual o produto permanece em chão-de fábrica.

**Tempo Setup**

- Tempo de preparação/adaptação de um recurso às novas condições.

**Tempo de Conclusão**

- Instante em que é finalizado o processamento de todas as tarefas.

**Restrição**

- Tudo o que limita a possibilidade de escolha.

**Produto**

- Resultado de um processo de fabrico, para ser comercializado de modo a satisfazer necessidades.

**Prioridade / Peso**

- Atributo de uma tarefa que define a preferência relativa da mesma em relação a outra em determinadas decisões.

**Data de disponibilidade**

- Data antes da qual o processamento não pode ser iniciado.

**Recurso**

- Máquinas ou matérias primas, ou outros elementos económicos utilizados em atividades de produção ou serviços.

**Tarefa (Job)**

- Conjunto de operações a concluir num determinado espaço de tempo.

## 2.4. Modelos de Sequenciamento

Os modelos de sequenciamento devem ser caracterizados pela interação das máquinas com os recursos disponíveis de acordo com as necessidades do processo. Esta interação ocorre no chão de fábrica (shop floor), onde é possível observar os seguintes ambientes de máquinas. [14]

- **Máquina única (Single Machine)**

Este é caracterizado por ser o mais simples de todos os ambientes. As tarefas apenas necessitam de uma operação em uma única máquina disponível, ou seja, as ordens de fabrico são processadas num único equipamento.

- **Máquinas paralelas (Parallel Machines)**

Existe um conjunto de *m* máquinas em que cada tarefa *j* só pode ser processada apenas por uma máquina, e cada máquina apenas poderá processar uma tarefa de cada vez. As máquinas poderão ser idênticas ou apresentar velocidades de processamento diferentes, assim como:

*Máquinas paralelas idênticas (Identical Machines), Pm*

Existem *m* máquinas idênticas em paralelo no qual uma tarefa *j* pode ser processada numa qualquer máquina e, o tempo de processamento da tarefa permanece constante independentemente da máquina a ser utilizada.

*Máquinas paralelas uniformes (Uniform machines), Qm*

Existem *m* máquinas que realizam o trabalho da mesma maneira diferindo apenas na velocidade de processamento. As máquinas podem ter características idênticas, mas com velocidade de processamento diferentes.

*Máquinas paralelas não relacionadas (Unrelated machines), Rm*

Máquinas sem nenhuma relação entre as velocidades de processamento. São caracterizadas quando existem *n* tarefas para serem repartidas entre *m* máquinas, mas cada tarefa é representada por um subconjunto contendo *t* tempos de execução. Os valores pertencentes ao subconjunto podem ser diferentes e representam o tempo de execução quando ela é atribuída a uma determinada máquina exclusivamente. [14]

- **Flow shop**

A sequência de processamento é comum a todas as ordens de fabrico. Cada tarefa *j* deverá ser processada em *m* máquinas em série. Todas as tarefas deverão seguir a mesma sequência, ou seja, passar por todas as *m* máquinas disponíveis. Este ambiente segue a regra FCFS (First Come First Served), ou seja, a primeira a chegar é a primeira a ser processada. [15]

- **Job shop**

Cada conjunto de tarefas têm uma sequência pré-determinada, sendo que poderão visitar a mesma máquina mais de que uma vez. O problema de Job-Shop consiste num conjunto de tarefas constituídas por um conjunto de operações que são realizadas em determinadas máquinas, sendo a ordem dessas operações específica das tarefas.

- **Open shop**

Existem *m* máquinas e cada tarefa terá de ser processada em cada uma das *m* máquinas. As ordens de produção podem ser processadas em qualquer sequência. Não há restrições quanto à ordem de realização das operações de cada tarefa nas diferentes máquinas ou na ordem pela qual máquina processa as diferentes tarefas, ou seja, a ordem das operações nas diferentes máquinas é irrelevante. [16]

- **Loteamento e Sequenciamento**

As ordens de produção são sequenciadas em lotes. Pode ser necessário definir a dimensão dos lotes e a sua sequência de fabrico. [5]

## 2.5. Job-shop scheduling

O problema da programação da produção/sequenciamento de operações é conhecido por Job-Shop Scheduling (JSS). Pode ser definido por *n* tarefas (*"job"*) a serem processadas no conjunto finito de *m* máquinas, com um determinado tempo de processamento e numa ordem pré-estabelecida, ou aleatoriamente sem qualquer comprometimento durante um determinado período temporal. Poderá haver mais do que uma máquina do mesmo tipo formando um centro de máquinas ("Shop"). [17]

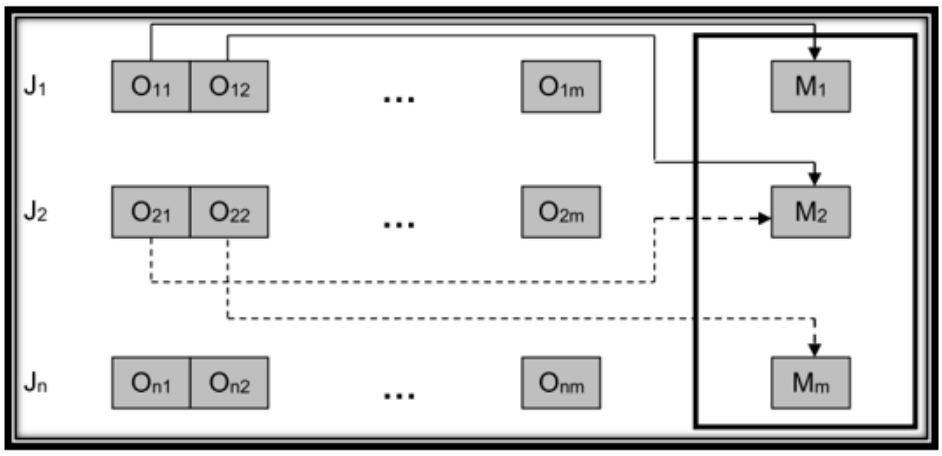


**Figura 6-Sequenciamento Job Shop. (Adaptado de Beirão, 1997)**

**De modo a que se poda simplificar o processo, são estabelecidos determinados pressupostos e condições , tais como:** [18]

- Todas as máquinas estão permanentemente disponíveis para processamento;
- Uma tarefa só pode ser executada por uma determinada máquina;
- Um processamento de uma tarefa tem de finalizar antes de iniciar a outra tarefa;
- Uma máquina apenas processa uma operação de cada vez e sem interrupção;
- Cada operação é processada numa única máquina durante um tempo fixo;
- A única limitação de recursos na oficina são as máquinas (mão-de-obra, materiais e ferramentas consideram-se sempre disponíveis).

O objetivo passa por atribuir todas as operações às máquinas, respeitando as restrições impostas, no menor tempo possível. [1]

### 2.5.1. Representação gráfica do problema.

Os métodos mais usuais para representar graficamente os problemas Job Shop são o Mapa de Grantt e o Grafo Disjuntivo.

#### 2.5.1.1. Mapa de Grantt.

Mapa de Grantt foi inicialmente introduzido por Henry L. Grantt no início da década de 1900. É uma ferramenta bastante utilizada no controlo da programação e sequenciamento de operações em ambiente do tipo JS, permitindo uma eficaz, intuitiva e rápida análise. É constituído por um sistema de eixos coordenados, em que o eixo das ordenadas é representado pelo conjunto de máquinas ou recursos, e o eixo das abcissas pelo tempo (normalmente em dias ou semanas). Segundo as regras, não poderão existir sobreposições de barras na mesma máquina, nem na mesma ordem de fabrico. [9]

A figura 7 representa graficamente um possível sequenciamento para 10 tarefas em 2 máquinas paralelas utilizando o Mapa de Grantt.

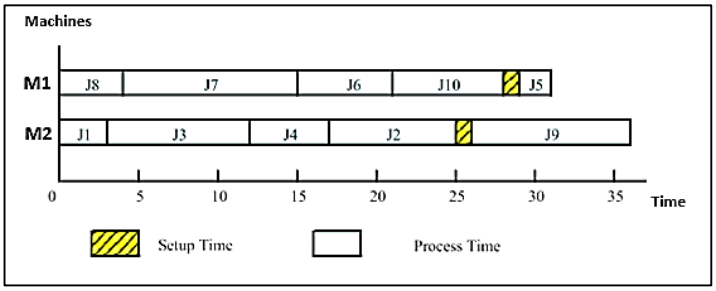


**Figura 7-Mapa de Grantt. (Adaptado de Mohamed. K. Omar, Siew C. Teo and Yasothei Suppiah)**

#### 2.5.1.2. Grafo Disjuntivo.

O Grafo Disjuntivo foi proposto por Roy e Sussman (1964). É um modelo de representação gráfica muito usado para descrever problemas de sequenciamento. É caracterizado por cada nó corresponder a uma determinada operação. Existem dois nodos adicionais, chamados I e F, que representam o início e o fim de um determinado sequenciamento. O instante de tempo da última operação sequenciada determina o valor do makespan ($C_{máx}$).

O problema pode ser definido por *n* tarefas e *m* máquinas num determinado chão de fábrica. A cada tarefa está associada uma determinada operação e cada máquina apenas pode processar uma operação de cada vez sem interrupção. O grafo disjuntivo contém todo o tipo de informação necessária para determinar uma solução (parcial ou completa) de um problema de sequenciamento job shop. A figura 8 ilustra o grafo disjuntivo aplicado ao problema de três ordens de fabrico por três máquinas.

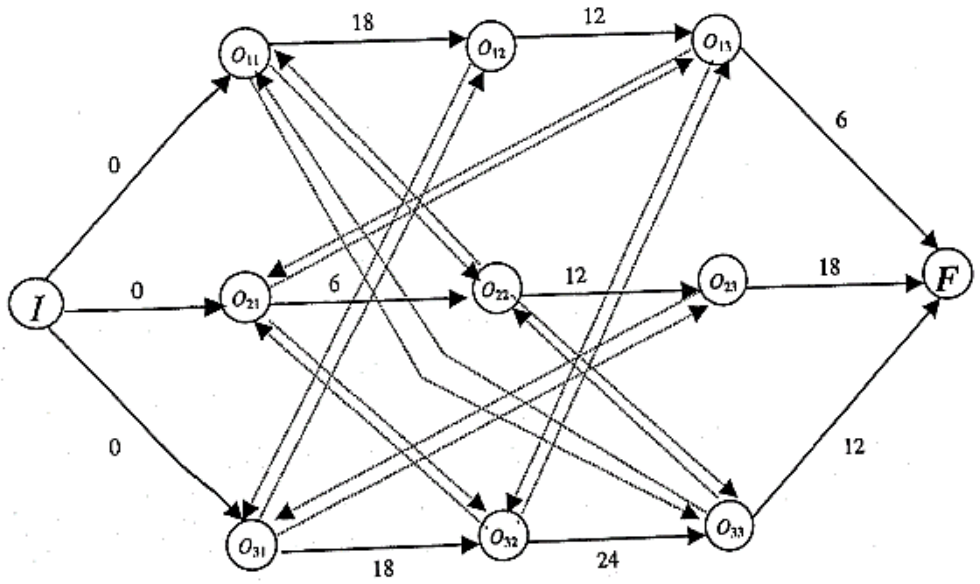


**Figura 8-Grafo Disjuntivo. (Beirão,1997)**

### 2.5.2. Regras de prioridade em ambiente Job Shop.

Para obter um determinado sequenciamento de um conjunto de tarefas é necessário definir prioridades, de modo a atingir um determinado objetivo. O sequenciamento obtido deverá ser tecnologicamente exequível, e acima de tudo eficiente, de acordo com o critério de desempenho definido. [5]

As regras de prioridade podem classificar-se quanto à sua variação ao longo do tempo e à informação que incorporam, podendo ser estáticas (as prioridades não variam ao longo do tempo), dinâmicas (as prioridades variam com o tempo), locais (apenas usam a informação da ordem de fabrico a que pertence), e globais (contêm informação de outras máquinas). [1]

Em seguida apresentam-se algumas das regras de prioridade mais comuns de acordo com Bou Shaala, A, Shouman, M. A, and Esheem, S. [20]

**PCO - "Preferred Customer Order".**

- A ordem de produção de um cliente considerado prioritário é processada primeiro.

**SPT - "Shortest Processing Time".**

- As operações são processadas por ordem crescente dos respetivos tempos de processamento. Permite uma redução das filas de espera.

**LPT - "Longest Processing Time".**

- As operações são processadas por ordem decrescente dos respetivos tempos de processamento.

**CR - “Critical Ratio”.**

- As ordens de fabrico devem ser processadas por ordem crescente de um determinado rácio. CR é definido pela diferença entre a data de entrega e o instante atual dividindo-se tempo de processamento restante.

**FCFS - "First Come First Served”.**

- As ordens de fabrico são processadas de acordo com a ordem de chegada ao centro de trabalho. A primeira a chegar é a primeira a ser processada. Esta regra visa minimizar o tempo de permanência na máquina, ou na fábrica.

**MINSOP - “Minimum Slack Time per Operation”.**

- Slack Time é definido pelo tempo restante até à data de entrega menos o tempo de processamento restante.

**EDD - "Earliest Due Date".**

- As ordens de fabrico são processadas por ordem crescente das datas de entrega (data prometida ao cliente). Esta regra prioriza as ordens mais urgentes, visando reduzir quer atrasos, quer o maior dos desvios entre as datas de conclusão e as datas devidas.

**LWKR - "Least Work Remaining".**

- A prioridade de processamento é dada à ordem de fabrico com o menor valor da soma das durações das operações ainda por realizar.

**MWKR - "Most Work Remaining".**

- A prioridade de processamento é dada à ordem de fabrico que tem o maior valor da soma das durações das operações ainda por realizar.

**RANDOM - "Random Selection".**

- Esta regra seleciona a próxima operação a ser processada, aleatoriamente.

## 2.6. Critérios de desempenho do sistema.

Para saber qual a regra de sequenciamento que tem melhor desempenho para um determinado objetivo, é necessário adotar critérios de desempenho. Critérios esses que serão aplicados ao nível da utilização dos recursos, do cumprimento dos prazos de entrega e, ao nível do stock. [5]

De seguida serão analisadas as medidas mais importantes referentes ao trabalho em questão.

- **Minimização do makespan.**

O makespan ($C_{máx}$) corresponde ao instante final da última ordem de fabrico, ou seja, é o tempo em que a última tarefa que está a ser processada na máquina termina.

- **Lateness**

Para um dado job *j*, a lateness é a diferença entre a data de entrega (*dj*) e a data de conclusão (*Cj*). Se for positiva, significa que a produção de uma tarefa acaba antes da data de entrega e, nesse caso, fala-se em earliness. Caso seja negativa, significa que se acaba a tarefa depois da data de entrega e, nesse caso, fala-se em tardiness.

## 2.7. Otimização combinatória

Os problemas de loteamento e sequenciamento em ambientes industriais enquadram-se numa classe de problemas de otimização combinatória, devido à sua elevada complexidade em termos computacionais. Um problema de otimização combinatória surge quando é necessário determinar a(s) solução(ões) ótima(s) em tempo considerado útil, respeitando um determinado conjunto de restrições.

Osman e Kelly referem que "os problemas de Otimização Combinatória são normalmente fáceis de descrever, mas difíceis de resolver". Referem ainda a otimização combinatória como "o estudo matemático para encontrar uma combinação, agrupamento, ordenação ou seleção ótima de objetos discretos usualmente finito em números". Cook (1971) desenvolveu a Teoria da Complexidade Computacional onde classificou a resolução dos problemas computacionais como de fácil (easy) ou difícil (hard). [13]

Um algoritmo é eficiente se a sua complexidade temporal crescer de forma polinomial e não de forma exponencial com a dimensão do problema. A eficiência é obtida através de soluções em tempo útil e a eficácia pela obtenção de soluções de boa qualidade. Perante isto, é possível dividir os problemas em três tipos de classes: [13]

- **P:** conjunto de problemas para os quais existe um algoritmo eficiente que cresce em tempo polinomial.

- **NP-Hard (Nondeterministic Polynomial time):** é um tipo de problema para o qual não existem algoritmos eficientes que o resolva em tempo polinomial, ou

seja, torna-se impossível testar todas as soluções possíveis em tempo computacional útil.

- **NP-Complete (Nondeterministic polynomial-time complete):** Subconjunto de problemas NP, em que caso seja encontrado um algoritmo polinomial, consequentemente existe um algoritmo polinomial para qualquer problema NP. Problemas considerados intratáveis, pertencem a esta classe.

## 2.8. Programação Linear.

Programação Linear (PL) é uma técnica matemática usada para resolver problemas de otimização num determinado modelo em que a função objetivo, bem como todas as restrições são lineares.

Quando todas as variáveis são discretas (pertencem ao conjunto de números inteiros) estamos perante um problema de Programação Linear Inteira (PLI). Caso as variáveis apresentem a condição de integralidade lidamos com um problema de Programação Linear Inteira Pura (PLIP). No entanto, se apenas algumas variáveis forem inteiras e outras contínuas, temos um problema de Programação Linear Inteira Mista (PLIM). Se as variáveis forem somente binárias (apenas assumem valores de 0 e 1), o modelo diz-se de Programação Linear Binária (PLB). [21]

## 2.9. GLPK (GNU Linear Programming Kit)

O GLPK é um kit de programação destinado a resolver problemas em larga escala (contendo um grande número de variáveis, na ordem de milhões), nomeadamente problemas de PL e PLIM. O GLPK usa o método simplex e o método do ponto interior para problemas não inteiros e o método Branch and Cut para problemas inteiros mistos. É um software livre que possui como principais componentes os seguintes métodos de resolução matemática: [22]

- Método Simplex Dual e Primal;
- Método do Ponto Interior Primal e Dual;

- Método “Branch and Cut”;
- Sistema Tradutor de Modelos do “AMPL” para o “GNU MathProg”;
- Interface para programas de aplicação;
- Sistema de resolução por linhas de comando, denomidado “stand-alone LP/MIP solver”.

A linguagem que este kit suporta é denominada por “GNU MathProg”, sendo um subconjunto da linguagem AMPL. Esta linguagem de modelagem matemática foi desenvolvida por Robert Fourer, David Gay, e Brian Kernighan. [23]

O AMPL permite resolver problemas de pequenas e grandes dimensões de elevada complexidade, no entanto; torna-se bastante eficaz em problemas de pequena dimensão solucionando-os num tempo razoável, ou seja, sem exigir grande esforço computacional. A linguagem faz a distinção entre um determinado modelo genérico (ficheiros .mod) e do conjunto de dados (ficheiros .dat); no entanto, não é obrigatório a existência de dois ficheiros separados. [24]

Por padrão, a metodologia utilizada para a resolução do problema é o algoritmo “branch-and-cut”.

## 2.10. Algoritmo Branch-and-Cut

O algoritmo “branch and cut” é um dos métodos mais eficientes utilizados em programação inteira mista. É um método híbrido gerado pela associação do algoritmo “branch and bound “(ramificação e limitação) e o método de planos de corte, no qual permite uma maior eficiência na procura de uma solução viável para o problema. O conceito base que está subjacente à técnica é dividir e resolver, permitindo eliminar soluções inviáveis. [25]

O problema inicial, por vezes, é considerado extremamente difícil (NP -Hard). Assim sendo, o algoritmo “branch and cut” procura encontrar uma solução ótima a partir da geração de uma árvore de nós (Figura 9) que representam subproblemas do problema principal. Este método é composto por duas operações: [26]

**Branching**

Cria ramificações a partir de um nó inicial para se obter subproblemas que apresentem um conjunto de soluções viáveis de modo a serem solucionadas.

**Bounding**

Elimina soluções de baixa qualidade através da comparação dos limites superiores (LS) e inferiores (LI). Num problema de minimização, o limite superior determina um valor conhecido e viável da função objetivo (poderá não ser o valor ótimo) que servirá de comparação com outras soluções obtidas. Assim, o valor do limite inferior deverá ser sempre menor ou igual ao valor da função objetivo. Caso o valor do limite inferior seja pior que o do limite superior, este será imediatamente retirado da árvore de pesquisa. [27]

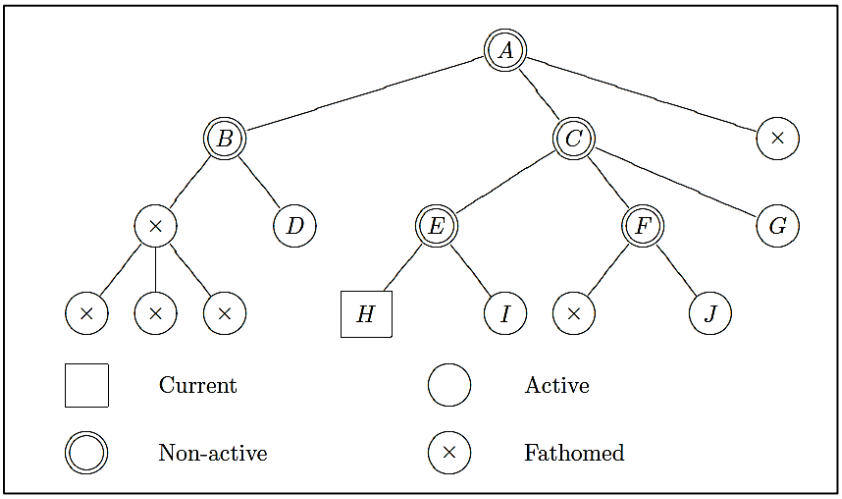


**Figura 9-Pesquisa em árvore. (Adaptado de GNU Linear Programming Kit, Reference Manual)**

Cada nó apresentado na árvore de pesquisa corresponde a um subproblema. Assim, podemos caracterizá-los da seguinte maneira:

**Currente (Actual) –** Nó que se encontra em processamento.

**Active (Activo) –** Nó que deverá ainda ser processado.

**Non-Active (Inativo) –** Nó que foi processado, mas não se encontra finalizado.

**Fathomed (Resolvido) –** Nó que foi processado e escolhido para sair da árvore de pesquisa.

# 3. DESENVOLVIMENTO DO TRABALHO

## 3.1. Descrição do problema

Ao recorrermos a técnicas de loteamento e sequenciamento, proporcionamos ganhos significativos em eficiência na produção, ou seja, eficiência a nível financeiro, bem como, no tempo necessário na produção.

O modelo foi desenvolvido de modo a obter um sequenciamento que respeite as datas de entregas associadas às tarefas, com o objetivo de minimizar o atraso total (total tardiness), assim como o número de mudanças de molde.

O problema é caracterizado por no chão de fábrica (shop floor) existir duas máquinas paralelas idênticas com a função de processar *n* tarefas num determinado período de tempo sem interrupção. Cada tarefa possui um tempo de processamento $p_j$, uma determinada família de moldes $f_j$, e uma data de entrega $d_j$ (tabela 2). Sempre que seja necessário realizar o processamento de tarefas de uma família para a outra é necessário um tempo de setup $S_{jk}$, nomeadamente para preparar a máquina para a mudança de molde. Entre tarefas da mesma família ($f_j = f_{k,}$), que apresentam características em comum, não é necessário nenhum tempo de adaptação/preparação $(S_{jk} = 0)$.

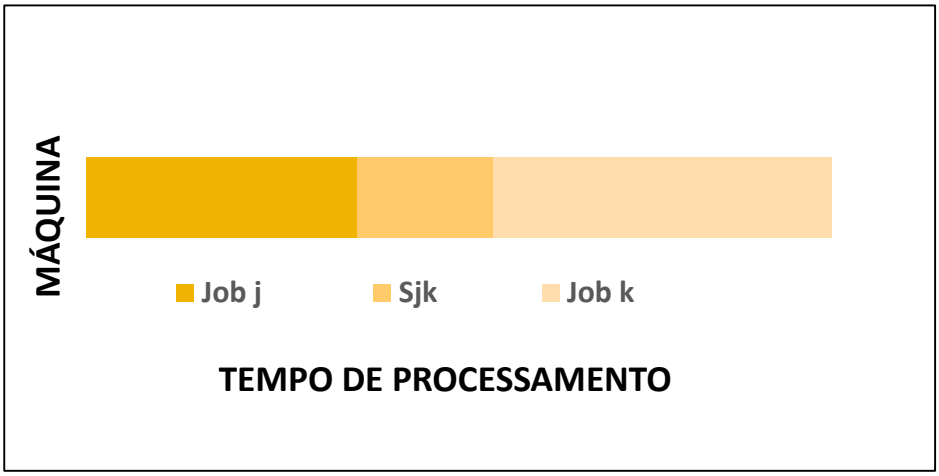


**Figura 10-Tempos de processamento**

O problema pode ser caracterizado pelas seguintes condições:

- As duas máquinas estão permanentemente disponíveis para processamento, desde que não estejam ocupadas.
- O processamento de uma tarefa tem de finalizar antes de iniciar a outra tarefa e, consequentemente, uma tarefa só pode ser executada por uma determinada máquina.
- Uma máquina apenas processa uma operação de cada vez e sem interrupção e durante um período tempo temporal.

### 3.1.1. Instância 1

Neste caso em específico, o problema foi testado considerando o seguinte:

- A família 5 apenas pode ser processada na máquina 1;
- A família 2, 3, 4, 9 e 10 apenas podem ser processadas na máquina 2;

Sempre que ocorre mudança de família, a máquina necessitará de um tempo de setup ($S_{jk}$) que corresponde à soma da desmontagem da família do molde onde está a ocorrer o processamento e a montagem da família do molde que será o próximo a ser processado.

### 3.1.2. Instância 2

Neste caso em específico, o problema foi solucionado considerando o seguinte:

- A família 1 pode ser processada em ambas as máquinas;
- A família 2 só pode ser processada na máquina 2;
- A família 3 só pode ser feita na máquina 1.

Quando ocorre mudança de família, a máquina necessitará de um tempo de setup equivalente a 1 hora.

**Tabela 2-Características das tarefas da Instância 2.**

| Tarefas | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| **Família** | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 3 | 3 | 3 |
| **Tempo processamento** | 3 | 8 | 9 | 5 | 2 | 6 | 11 | 4 | 10 | 7 |
| **Data de Entrega** | 11 | 18 | 16 | 17 | 27 | 19 | 15 | 12 | 21 | 26 |
| **Tempo de Setup** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

## 3.2. Modelo Matemático

Neste caso em específico o problema passa pela minimização da função objetivo em que se procura o menor valor possível sem infringir nenhuma restrição. Na abordagem ao problema de otimização foi implementado um modelo matemático de programação linear inteira mista (PLIM). Este é uma adaptação do modelo proposto por Mohamed. K. Omar, Siew C. Teo e Yasothei Suppiah. [8]

### 3.2.1. Parâmetros

- $m$ = número de famílias de moldes.
- $n$ = número total de tarefas.
- $r$ = número de máquinas.
- $f_j$ = família de moldes referente à tarefa $j$, $f_j = 1, 2, \ldots, m$
- $d_j$ = data de entrega da tarefa $j$.
- $p_j$ = tempo de processamento da tarefa $j$ na máquina $b$.
- $S_{jk}$ = Tempo de setup da família de moldes da tarefa $j$ para a família de moldes da tarefa $k$.
- $M_{jk} = \begin{cases} S_{jk} \ se \ f_j \neq f_k \\ 0 \ se \ f_j = f_k \end{cases}$
- $G$ = Número inteiro suficientemente grande.

### 3.2.2. Variáveis de Decisão

- $C_j$ = conclusão da tarefa j
- $T_j$ = atraso da tarefa j
- $\alpha_{jb} = \begin{cases} 1 \text{ se a tarefa } j \text{ for a primeira a ser processada na máquina } b \\ 0 \text{ caso contrário} \end{cases}$
- $\theta_{jk} = \begin{cases} 1 \text{ se a tarefa } k \text{ for agendada logo após a tarefa } j. \\ 0 \text{ caso contrário} \end{cases}$
- $\beta_{jb}$ Variável contínua restrita ao intervalo [0, 1], indicando que a tarefa *j* foi agendada na máquina *b*, mas não em primeiro lugar.

### 3.2.3. Formulação Matemática

*Minimizar:*

$$\sum_{j=1}^{n} T_j \tag{1}$$

*Sujeito a:*

$$\sum_{b=1}^{r} (\alpha_{jb} + \beta_{jb}) = 1 \ , j = 1, 2, \dots, n \tag{2}$$

$$f_j = 3, \quad so \sum_{b=1}^{r} (\alpha_{j1} + \beta_{j1}) = 1 \ , j = 1, 2, \dots, n \tag{2x}$$

$$f_j = 2, \quad so \sum_{b=1}^{r} (\alpha_{j2} + \beta_{j2}) = 1 \ , j = 1, 2, \dots, n \tag{2y}$$

$$\alpha_{jb} + \beta_{jb} \leq \beta_{kb} + 1 - \theta_{jk} \ , j = 1, 2, \dots, n; k = 1, 2, \dots, n; \ b = 1, 2, \dots, r \tag{3}$$

$$\sum_{j=1}^{n} \alpha_{jb} \leq 1 \ , b = 1, 2, \dots, r \tag{4}$$

$$\sum_{b=1}^{r} \alpha_{jb} + \sum_{k=1}^{n} \Theta_{kj} = 1 \ , j = 1, 2, \dots, n \tag{5}$$

$$\sum_{k=1}^{n} \Theta_{jk} \leq 1 \ , j = 1, 2, \dots, n \tag{6}$$

$$C_k \geq \ C_j + M_{jk} + \sum_{b=1}^{r} p_k \beta_{kb} + G\Theta_{jk} - G \ \ , j = 1, 2, \dots, n; \ , k = 1, 2, \dots, n \tag{7}$$

$$C_j \geq \sum_{b=1}^{r} p_j \left(\alpha_{jb} + \ \beta_{jb}\right), j = 1, 2, \dots, n \tag{8}$$

$$C_j \ \leq \ d_j + \ T_j \ \ , j = 1, 2, \dots, n \tag{9}$$

$$d_j \Theta_{jk} \leq \ d_k \Theta_{jk} \ \ , j = 1, 2, \dots, n; \ k = 1, 2, \dots, n \tag{10}$$

$$C_j \ , T_j \ \ \geq 0 \ , j = 1, 2, \dots, n \tag{11}$$

A equação (1) representa a função objetivo, que minimiza a soma total dos atrasos das tarefas. A equação (2) assegura que cada tarefa deverá ser atribuída a uma determinada máquina e não a duas ao mesmo tempo. A equação (2x) determina que as tarefas da família 3 deverão ser processadas na máquina 1. A equação (2y) determina que as tarefas da família 2 deverão ser processadas na máquina 1.

A equação (3) obriga a que as tarefas e os respetivos sucessores da sequência de processamento sejam processados na mesma máquina. A equação (4) garante que cada tarefa apenas poderá ser processada numa determinada máquina. A equação (5) obriga a que uma tarefa seja a primeira a ser processada ou seja procedida por outra na sequência de processamento. A equação (6) certifica que cada tarefa deve ser sucedida por outra. A equação (7) assegura que o tempo de arranque de uma tarefa nunca pode ser inferior ao tempo total de finalização da encomenda anterior.

A equação (8) obriga a que o tempo de finalização de uma tarefa seja superior ou igual ao tempo de processamento. A equação (9) mede o grau pela qual uma tarefa sofre atraso. A equação (10) força que a data de entrega de uma tarefa seja a mesma ou inferior à do seu direto sucessor. A equação (11) assegura que os valores deverão ser positivos.

# 4. ANÁLISE E DISCUSSÃO DOS RESULTADOS

Para determinação dos resultados foi utilizado um computador portátil com as seguintes características:

- CPU: Intel Core I5-3317U, 1.7 GHz.
- Memória Ram: 4 GB.
- Sistema Operativo Windows 7 Ultimate 64bits.

O Software usado para resolver o problema foi o GLPK Integer Optimizer, versão 4.60. Contém um solver (glpsol.exe) que é invocado diretamente na interface de linha de comandos (cmd.exe), onde é possível manusear, visualizar e manipular arquivos no computador.

Um problema NP-Hard leva imenso tempo a ser resolvido, então, de modo a podermos acompanhar a resolução de um determinado modelo, o solver descreve no terminal as informações no seguinte formato:

- **+nnn: mip = xxx <rho> yyy gap (ppp; qqq)**

Numa análise mais detalhada das informações inerentes ao formato acima mencionado, temos:

- **"nnn" -** relata o número de interações que já foram solucionadas.
- **"mip" –** significa que foi encontrada uma solução para o problema.
- **"xxx"** – indica o valor da função objetivo.
- **"<rho>"** – é a mensagem que nos indica se o problema é de minimização (>=) ou de maximização (<=).
- **"yyy" –** limite global para o melhor valor inteiro.

- **"gap" -** valor em percentagem que diminui (em caso de minimização) proporcionalmente com o decorrer da resolução do problema. O ficheiro de saída é obtido apenas quando a percentagem é zero.
- **"ppp" -** número de subproblemas que se encontram na lista à espera de resolução.
- **"qqq" –** número de subproblemas que foram resolvidos e retirados da árvore de pesquisa.

Em seguida, podemos observar dois casos específicos que foram analisados ao longo deste trabalho.

## 4.1. Instância 1

Inicialmente realizou-se vários testes para um ambiente de 32 tarefas e 2 máquinas. Para que fosse possível analisar o problema foi necessário digitar o seguinte comando:

- **C:\Users\JoãoVeríssimo\winglpk-4.60\glpk-4.60\w64>glpsol.exe -m C:\glp-works\Model32Jobs.mod**

Devido à elevada quantidade de informação (Anexo A), verificou-se que o solver não consegue solucionar o problema num tempo computacional útil. Na figura 11 verifica-se que existe uma ineficácia em tempo computacional. Durante 64516.3 segundos (cerca de 18 horas) o número de subproblemas continua a aumentar e a percentagem tende a estagnar, tornando-se inviável ao suporte na tomada de decisão em ambiente industrial.

Perante estas conclusões, optou-se por analisar a instância 2 (Anexo B), que possui uma menor quantidade de informação.

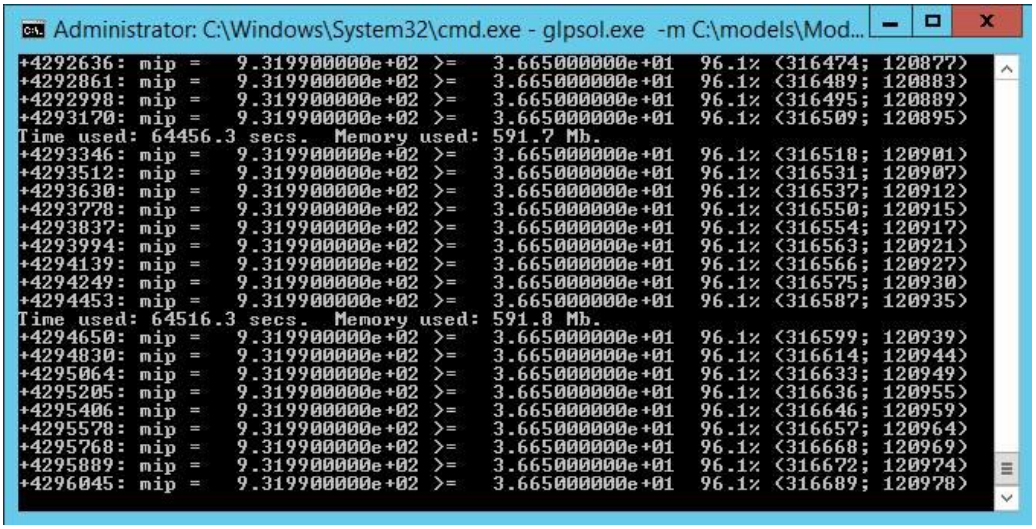


**Figura 11-32Jobs**

## 4.2. Instância 2

Neste caso foi possível testar o modelo para 10 tarefas e 2 máquinas. Constatou-se que a equação (10) interfere drasticamente no desenrolar do processo, ou seja, a sua existência permite acelerar a convergência, e encontrar rapidamente uma solução para o problema.

Serão apresentados e comparados os resultados para dois casos relativos à mesma instância. No caso 1 a equação 10 está presente no modelo e será gerada com sucesso (figura 12), e no caso 2 abdicou-se da equação 10, conforme se pode comprovar na figura 19.

### 4.2.1. Caso 1

De modo a solucionar o problema e obter um ficheiro de saída (.txt) foi necessário escrever a seguinte linha de comandos:

- **C:\Users\JoãoVeríssimo\winglpk-4.60\glpk-4.60\w64>glpsol.exe -m C:\glp-works\Jobs.mod -o Jobs.txt**

De acordo com a linha de comandos acima, a opção -m filename ou --model filename vai permitir a leitura do modelo Jobs.mod, e se for o caso, o ficheiro de dados

(filename.dat). A opção -o filename ou --output filename possibilita a obtenção de um ficheiro de saída, neste caso Jobs.txt. Na figura 12 podemos observar que a linha de comandos foi aceite com sucesso e o modelo foi resolvido, originando então um ficheiro de saída (figura 13) para a análise do problema em questão. [25]

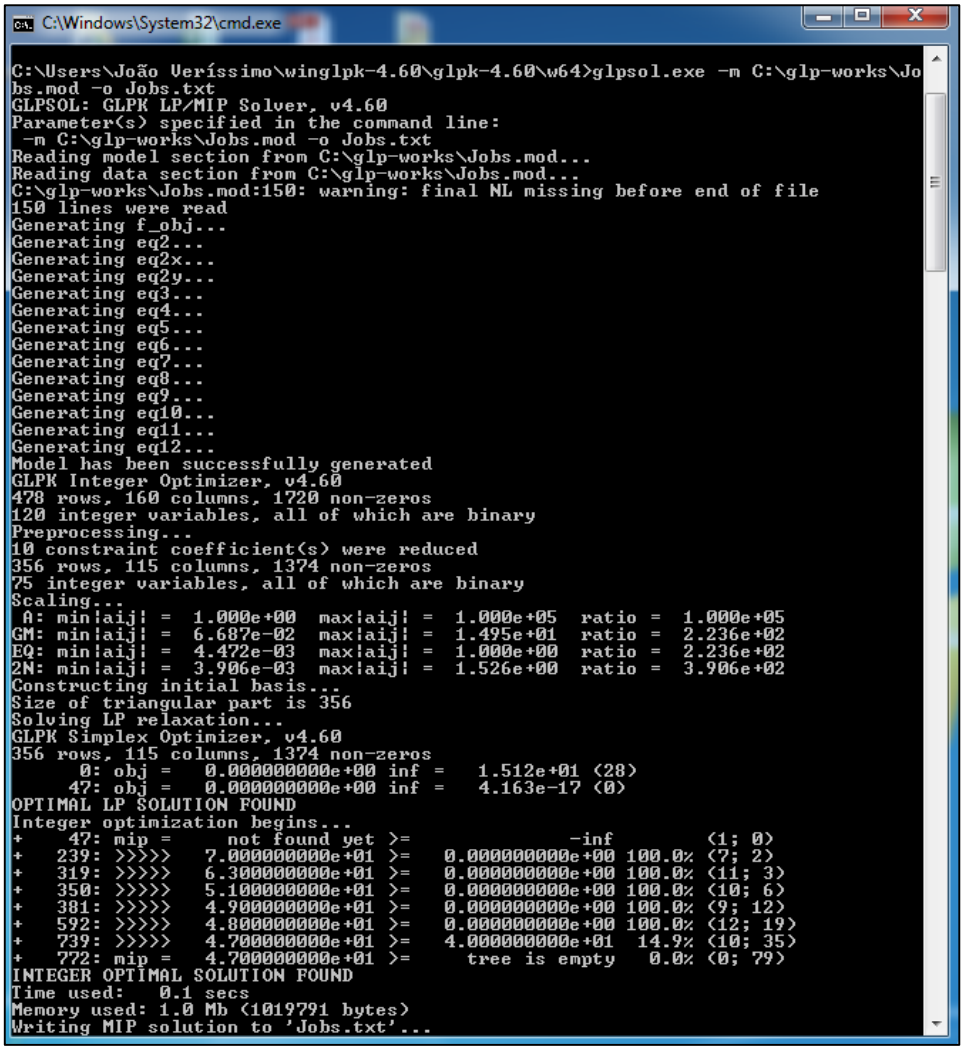


**Figura 12-Linha de comandos Jobs.mod**

```
Problem:    Jobs
Rows:       478
Columns:    160 (120 integer, 120 binary)
Non-zeros:  1720
Status:     INTEGER OPTIMAL
Objective:  f_obj = 47 (MINimum)
```

**Figura 13-Ficheiro de saída Jobs.txt**

Ao analisar a figura 12 podemos observar que a aplicação percorreu todas as linhas do modelo, executando com sucesso todas as equações, encontrando uma solução ótima inteira para a função objectivo (não significa que é a solução ótima).

Perante isto, podemos observar que ocorreu cerca de 772 interações. Foi encontrado o valor da função objetivo ($4.7 \times 10^1$) horas, num total de 0.1 segundos. Ao longo do tempo a informação acumulada foi cerca de 1019791 bytes (1 MB) e verifica-se que foram descartadas cerca de 79 subproblemas até ser encontrada uma solução óptima para a função objectivo.

Após uma análise do ficheiro output é possível construir a tabela 3:

**Tabela 3-Tabela de Resultados Caso 1**

| Máquina | Tarefa | Molde | Tempo Processamento (h) | Tempo Conclusão (h) | Setup Time (h) | $C_{máx}$ (h) |
|---|---|---|---|---|---|---|
| 1 | **8** | 3 | 4 | 4 | 1 | **115** |
| | **3** | 1 | 9 | 14 | 0 | |
| | **2** | 1 | 8 | 22 | 1 | |
| | **9** | 3 | 10 | 33 | 0 | |
| | **10** | 3 | 7 | 40 | 0 | |
| 2 | **1** | 1 | 3 | 3 | 1 | **102** |
| | **7** | 2 | 11 | 15 | 1 | |
| | **4** | 1 | 5 | 21 | 1 | |
| | **6** | 2 | 6 | 28 | 1 | |
| | **5** | 1 | 2 | 31 | 0 | |

Em suma, podemos observar que todas as restrições foram respeitadas, ou seja, as tarefas correspondentes à família do molde 2 foram processadas na máquina 2 e as tarefas referentes à família do molde 3 foram processadas na máquina 1. Verificamos ainda que na transição de tarefas em que a família de moldes é diferente, existe tempo de setup, ou seja, o tempo necessário para adapatação da máquina de modo a que esta possa processar a tarefa seguinte.

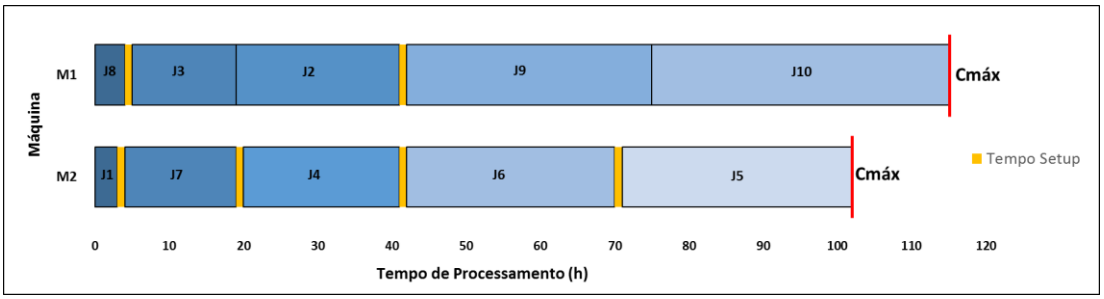


**Figura 14-Sequenciamento das Tarefas Caso 1**

Para uma análise mais rápida e intuitiva, podemos observar o Mapa de Grantt, onde é possível verificar rapidamente o sequenciamento ao longo do tempo nas diversas máquinas, assim como o tempo de setup entre tarefas de diferentes famílias moldes.

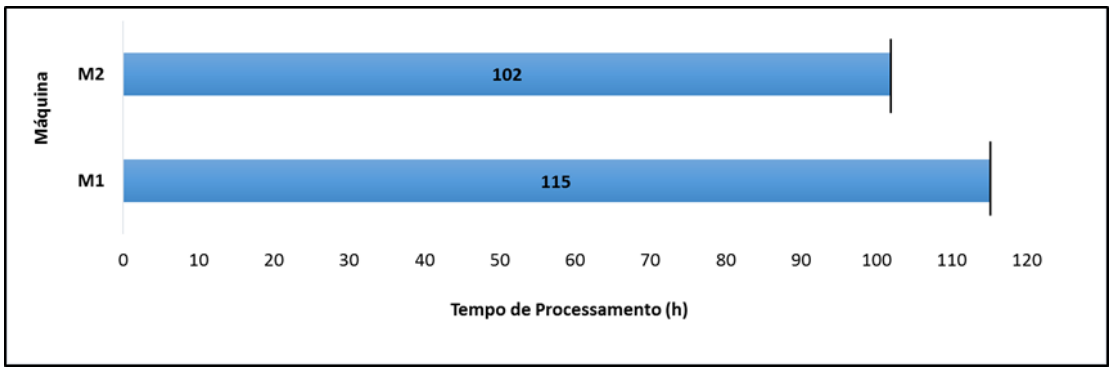


**Figura 15-Makespan ($C_{máx}$) Caso1**

Podemos ainda verificar que a soma total dos tempos de processamento das tarefas com o tempo de setup é de 115 horas na máquina 1 e 102 horas na máquina 2. O tempo total de processamento ($C_{máx}$) é determinado quando a última tarefa abandona o sistema, o qual também designamos por makespan.

**Tabela 4-Lateness Caso 1**

| Tarefas | Data de Conclusão (h) | Data de Entrega (h) | Lateness (h) |
|---|---|---|---|
| 1 | 3 | 11 | **8** |
| 2 | 22 | 18 | **-4** |
| 3 | 14 | 16 | **2** |
| 4 | 21 | 17 | **-4** |
| 5 | 31 | 27 | **-4** |
| 6 | 28 | 19 | **-9** |
| 7 | 15 | 15 | **0** |
| 8 | 4 | 12 | **8** |
| 9 | 33 | 21 | **-12** |
| 10 | 40 | 26 | **-14** |
| ***Adiantamento Total (h)*** | | | **18** |
| ***Atraso Total (h)*** | | | **-47** |

Ao observar a tabela 4, podemos de uma forma objetiva e individual identificar qual das tarefas cumpre a data de entrega de acordo com o sequenciamento obtido anteriormente.

O sequenciamento obtido segue a regra de prioridade EDD, ou seja, dá prioridade às tarefas com mais urgência de modo a cumprir o prazo de entrega. No entanto, devido às restrições do modelo, o melhor valor obtido para o atraso total das 10 tarefas foi de 47 horas.

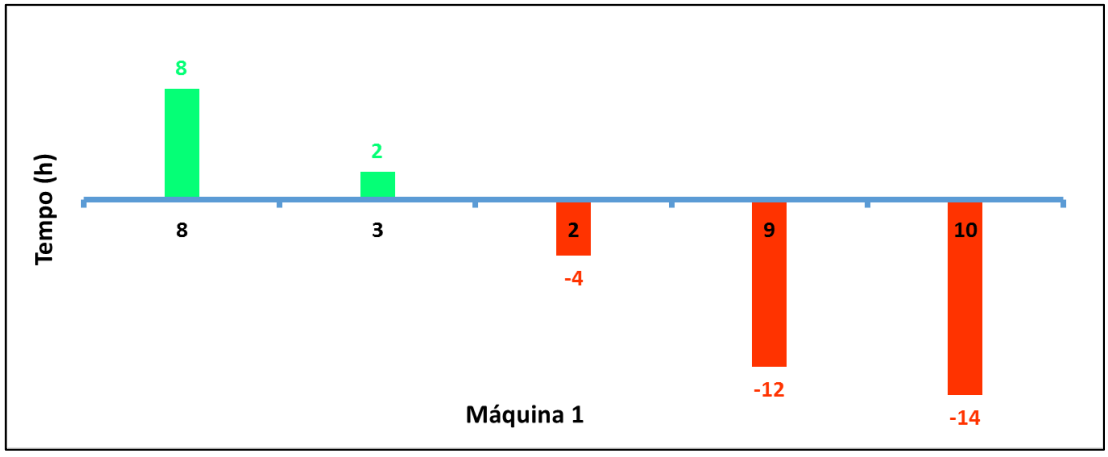


**Figura 16-Lateness Máquina 1 Caso 1**

O processamento das tarefas na máquina 1 é concluido ao fim de 115 horas de trabalho e a soma total do atraso das encomendas é de 30 horas. A conclusão da tarefa 8 termina 8 horas mais cedo que a data de entrega prevista e a tarefa 3 finaliza 2 horas mais cedo.

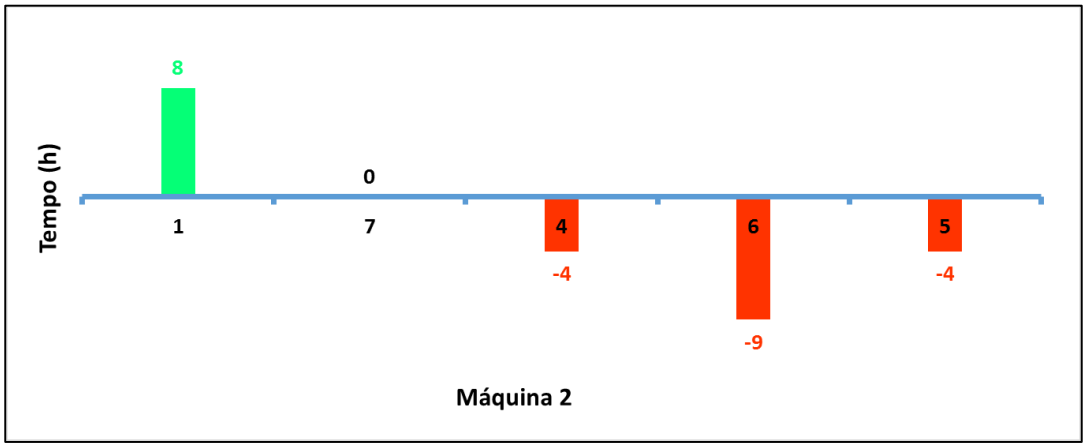


**Figura 17-Lateness Máquina 2 Caso 1**

Relativamente ao processamento das tarefas na máquina 2, este apresenta um valor total de 102 horas de trabalho e a soma total do atraso das encomendas é de 17 horas.

A conclusão da tarefa 1 termina 8 horas mais cedo que a data de entrega prevista e a tarefa sete termina justamente no tempo exacto da data de entrega da encomenda.

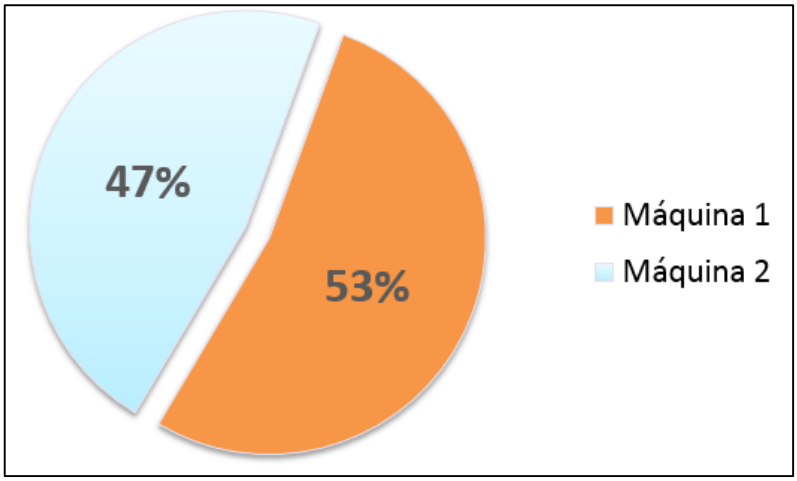


**Figura 18-Ocupação das Máquinas Caso 1**

Podemos ainda verificar que a máquina 1 esteve a trabalhar um pouco mais do que a máquina 2, apresentando um valor de 53% da ocupação total, contra os 47% da máquina 2.

### 4.2.2. Caso 2

Através da seguinte linha de comandos foi possível obter o ficheiro de saída (Jobs_1.txt) alusivo ao caso 2:

- **C:\Users\JoãoVeríssimo\winglpk-4.60\glpk-4.60\w64>glpsol.exe -m C:\glp-works\Jobs_1.mod -o Jobs_1.txt**

```
147 lines were read
Generating f_obj...
Generating eq2...
Generating eq2x...
Generating eq2y...
Generating eq3...
Generating eq4...
Generating eq5...
Generating eq6...
Generating eq7...
Generating eq8...
Generating eq9...
Generating eq11...
Generating eq12...
Model has been successfully generated
GLPK Integer Optimizer, v4.60
378 rows, 160 columns, 1630 non-zeros
120 integer variables, all of which are binary
Preprocessing...
10 constraint coefficient(s) were reduced
357 rows, 160 columns, 1600 non-zeros
120 integer variables, all of which are binary
Scaling...
 A: min|aij| =  1.000e+00  max|aij| =  1.000e+05  ratio =  1.000e+05
GM: min|aij| =  6.687e-02  max|aij| =  1.495e+01  ratio =  2.236e+02
EQ: min|aij| =  4.472e-03  max|aij| =  1.000e+00  ratio =  2.236e+02
2N: min|aij| =  3.906e-03  max|aij| =  1.526e+00  ratio =  3.906e+02
Constructing initial basis...
Size of triangular part is 357
Solving LP relaxation...
GLPK Simplex Optimizer, v4.60
Time used: 55048.8 secs.  Memory used: 602.4 Mb.
+5837526: mip =   3.400000000e+01 >=   3.300000000e+01   2.9% (5638; 2158397)
Warning: numerical instability (dual simplex, phase II)
+5847187: mip =   3.400000000e+01 >=   3.300000000e+01   2.9% (2418; 2188834)
+5859700: mip =   3.400000000e+01 >=     tree is empty   0.0% (0; 2217463)
INTEGER OPTIMAL SOLUTION FOUND
Time used:   55061.8 secs
Memory used: 602.5 Mb (631747914 bytes)
```

**Figura 19-Jobs_1.mod**

Neste caso verifica-se um total 5859700 interações em 55061.8 segundos (cerca de 15 horas e 17 minutos) até obter uma solução ótima ($3.4 \times 10^1$). Ao longo do tempo computacional foram resolvidos 2217463 subproblemas, e uma informação acumulada de 602.5Mb.

**Tabela 5-Tabela de Resultados Caso 2**

| Máquina | Tarefas | Molde | Tempo Processamento (h) | Tempo Conclusão (h) | Setup Time (h) | $C_{máx}$ (h) |
|---|---|---|---|---|---|---|
| | **8** | *3* | 4 | *4* | 1 | |
| | **4** | *1* | 5 | *10* | 0 | |
| ***1*** | **2** | *1* | 8 | *18* | 1 | ***96*** |
| | **10** | *3* | 7 | *26* | 0 | |
| | **9** | *3* | 10 | *36* | 0 | |
| | **1** | 1 | 3 | 3 | 1 | |
| | **7** | 2 | 11 | 15 | 0 | |
| ***2*** | **6** | 2 | 6 | 21 | 1 | **98** |
| | **5** | 1 | 2 | 24 | 0 | |
| | **3** | 1 | 9 | 33 | 0 | |

Após a análise do ficheiro de saída, verifica-se que todas as restrições foram cumpridas. Em comparação com o caso 1 verifica-se que o Makespan ($C_{máx}$) é inferior. Na máquina 1 o valor decresce de 115 para 96 horas e, na máquina 2 de 102 para 98 horas. Tal facto deve-se à diminuição da troca de moldes, e consequentemente a geração de uma nova sequência de produção (Figura 20).

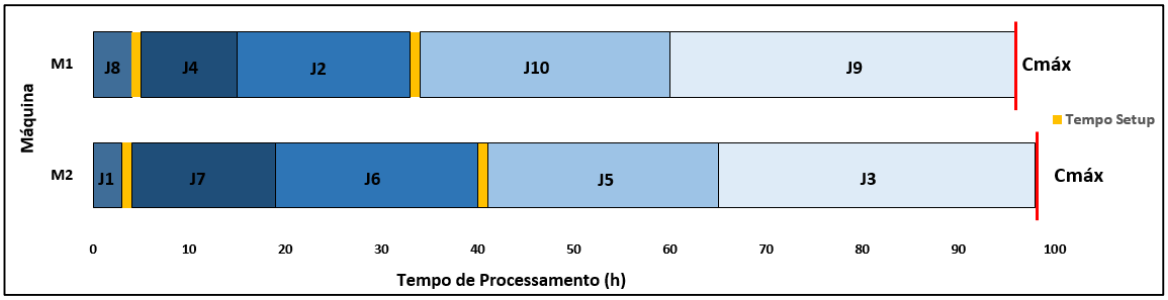


**Figura 20-Sequenciamento das Tarefas Caso 2**

Uma das alterações foi a troca da tarefa 4 para a máquina 1 e da tarefa 3 para a máquina 2. Verifica-se também uma permuta entre a tarefa 9 e 10 na máquina 1, e na máquina 2 as tarefas 5 e 6 iniciam mais cedo.

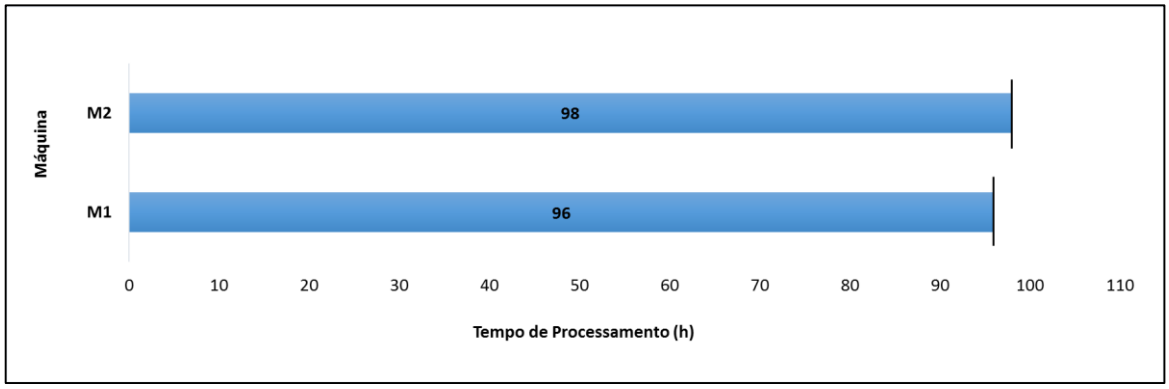


**Figura 21-Makespan ($C_{máx}$) Caso 2**

Devido ao novo sequenciamento da produção verificamos que a máquina 1 termina o processamento das tarefas mais cedo (cerca de 19 horas) em relação ao caso 1, e 2 horas em relação à máquina 2. A máquina 2 termina 4 horas mais cedo em relação ao caso 1.

**Tabela 6-Lateness Caso 2**

| Tarefas | Data de Conclusão (h) | Data de Entrega (h) | Lateness (h) |
|---|---|---|---|
| 1 | 4 | 11 | **8** |
| 2 | 18 | 18 | **0** |
| 3 | 33 | 16 | **-17** |
| 4 | 10 | 17 | **7** |
| 5 | 24 | 27 | **3** |
| 6 | 21 | 19 | **-2** |
| 7 | 15 | 15 | **0** |
| 8 | 4 | 12 | **8** |
| 9 | 36 | 21 | **-15** |
| 10 | 26 | 26 | **0** |
| | ***Adiantamento (h)*** | | **26** |
| | ***Atraso Total (h)*** | | **-34** |

Neste caso a soma do atraso total das tarefas a serem processadas foi reduzido de 47 para 34 horas. Para a minimização do atraso total, ocorreu uma otimização do processo, de modo a cumprir ou melhorar a data de entrega. A tarefa 6 apesar de não cumprir a data de entrega, melhorou em 7 horas a sua data de conclusão. As tarefas 3 e 9, são a únicas que

demoram mais tempo a serem concluídas em relação ao caso 1, e consequentemente as últimas a serem concluídas.

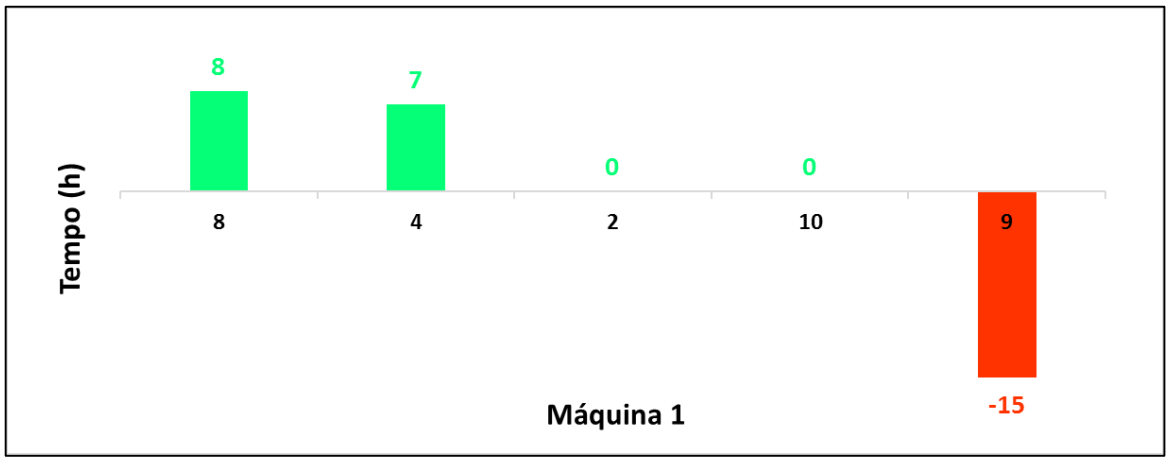


**Figura 22-Lateness Máquina 1 Caso 2**

Na máquina 1 o atraso total é de 15 horas, contribuindo para este resultado apenas e unicamente tarefa 9.

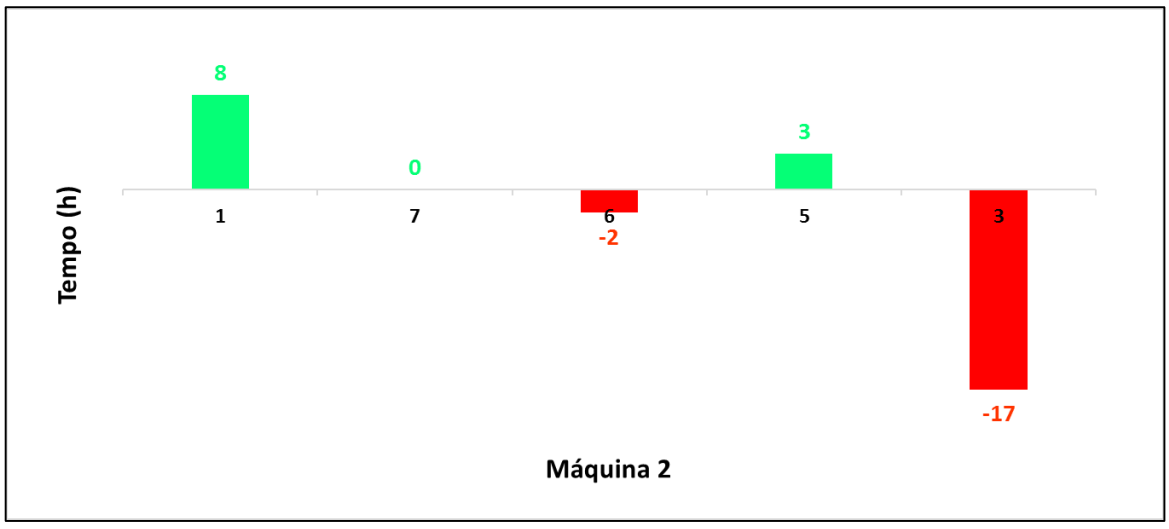


**Figura 23-Lateness Máquina 2 Caso 2**

Na máquina 2 o atraso total é de 19 horas, contribuindo para este resultado a tarefa 6 e 9.

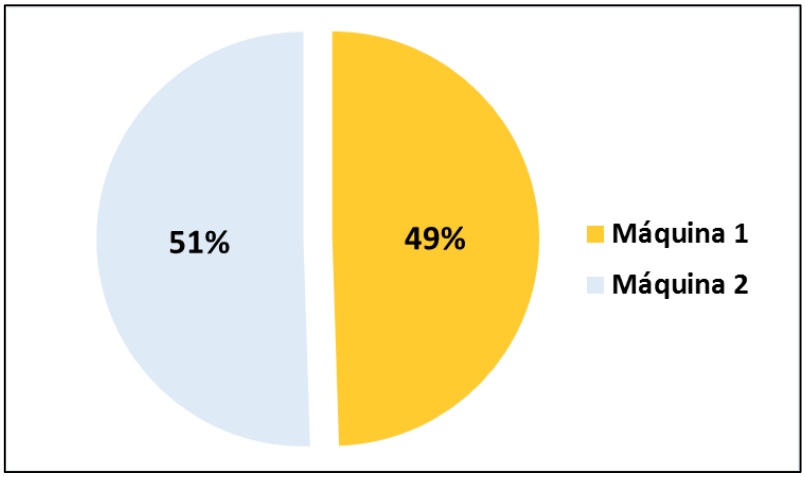


**Figura 24-Ocupação das Máquinas Caso 2**

Enquanto que no caso um havia uma diferença percentual de 4% entre as respetivas máquinas, agora a diferença diminuiu para 1%.

Em suma, podemos concluir que a utilização da equação 10, faz com que no momento da pesquisa em árvore, o solver passe ao lado de uma solução melhor. No caso 2 obteve-se resultados significativamente melhores. O tempo computacional foi mais elevado devido à quantidade de subproblemas que foram criados até se encontrar uma solução ótima.

# 5. CONCLUSÕES

O objetivo deste trabalho foi a realização de estudo sobre problemas de loteamento e sequenciamento de operações do tipo job shop, em ambiente de máquinas paralelas idênticas, e com os tempos de processamentos dependentes das tarefas. Foi usado um modelo matemático PLIM, implementado na linguagem de modelagem matemática AMPL com recurso ao software GLPK.

O critério de desempenho atraso total (total tardiness) tem uma elevada importância nos sistemas de produção e pode acarretar elevados custos, nomeadamente quando existe um atraso na entrega de uma tarefa. Pode ocorrer aplicação de multas, perda de credibilidade e consequentemente perda de clientes e ainda diminuição da reputação da empresa. Em mercados competitivos isto torna-se inaceitável e, para isso, é necessário otimizar os processos, racionalizando os recursos, com foco na diminuição do atraso total das tarefas.

Anteriormente vimos que, ao aplicarmos técnicas de loteamento e sequenciamento, existe uma maior eficiência na programação da produção, mas também na diminuição dos recursos financeiros, assim como na diminuição do tempo necessário para processamento das tarefas. Podemos observar que um dos principais problemas com que nos deparamos é quando ocorre a mudança de molde após a conclusão de uma tarefa, surgindo assim o tempo de setup, o que irá atrasar o $C_{máx}$.

Relativamente à instância 2, concluímos que a presença da equação 10 acelera a convergência do modelo. Pela análise da árvore de pesquisa verificamos que no caso 1 foram resolvidos cerca de 79 subproblemas em cerca de 0.1 segundos, enquanto que, no caso 2, foram resolvidos cerca de 2217463 subproblemas em 55048.8 segundos (cerca de 15 horas e 17 minutos). Consequentemente o atraso total no caso 1 foi 47 horas, e no caso 2 foi 34 horas.

Por definição, o GLPK solver utiliza o algoritmo Branch and Cut. Este torna-se bastante eficaz para resolver problemas de calibre menor (Anexo B), no entanto, para problemas de larga escala, ele não é eficaz. Em problemas com instâncias maiores (Anexo A), foi possível constatar uma ineficácia de resolução em termos de tempo computacional.

Devido à elevada natureza combinatória intrínseca a este tipo de problemas, torna-se inexequível testar todas as opções possíveis num tempo computacional aceitável.

Foi possível obter uma ferramenta para auxiliar na tomada de decisão por parte das empresas. Esta, permite-nos atingir com sucesso os objetivos iniciais delineados, tais como: um sequenciamento de *n* tarefas em *m* máquinas, o loteamento das tarefas nas *m* máquinas respeitando as famílias de moldes inerentes a cada máquina e, acima de tudo, a minimização do atraso total das tarefas ao longo do tempo.

Como trabalho futuro, pretende-se testar o modelo matemático em diversos tipos de software presentes no mercado (comerciais ou gratuitos), de forma a verificar a sua eficácia, tanto a nível de resultados como de tempo computacional.

Pretende-se ainda, adicionar a seguinte equação:

$$\sum_{b=1}^{r} \left( C_{f_{j,b}} \times \beta_{jb} \right) = 1 \quad , j = 1, 2, \dots, n \tag{13}$$

A equação (13) verifica a compatibilidade de uma determinada família de moldes numa dada máquina. Perante isto, as equações (2x) e (2y) serão substituídas pela equação acima.

# REFERÊNCIAS BIBLIOGRÁFICAS


1. Figueiredo J. Implementação de um Algoritmo Genético Híbrido com Simulated Annealing para o problema Job Shop. Universidade do Minho; 2015.
2. Silva C, Klement N, Gibaru O. A generic decision support tool for lot-sizing and scheduling problems with setup and due dates. Int Jt Conf. 2016;7296.
3. Moustakis V (University of C. Material Requirements Planning (MRP). Technical University of Crete; 2000.
4. Kolharkar S, Zanwar D, Kawade G. Scheduling in Job Shop Process Industry. Iosr-Jmce. 2013;5(1):1–17.
5. Silva C. Sequenciamento. Coimbra; 2008. p. 1–14.
6. Toyota-Global. Just in Time (Kanban). www.toyota-global.com. 1995.
7. Udaiyakumar KC, Chandrasekaran M. Application of firefly algorithm in job shop scheduling problem for minimization of Makespan. Procedia Eng. 2014;97:1798–807.
8. Omar MK, Teo SC, Suppiah Y. Scheduling with Setup Considerations : An MIP Approach. Multiprocessor Sched Theory Appl. 2007;(December):436.
9. CEFAMOL. A Indústria Portuguesa de Moldes. 2015;2–8.
10. Silva C, Ferreira L. Microplano – A Scheduling Support System for the Plastic Injection Industry. 2002;
11. Vilar Jacob V. Aplicação de Metaheurísticas para problemas de sequenciamento com lotes de tarefas. Universidade Federal de Viçosa; 2014.
12. Ouelhadj D, Petrovic S. A survey of dynamic scheduling in manufacturing systems. J Sched. 2009;12(4):417–31.
13. Soares I. Sistema Inteligente para Escalonamento Assistido por Aprendizagem. Universidade de Trás-os-Montes e Alto Douro Sistema; 2014.
14. Galvão FM. Aplicação de um modelo de sequenciamento da produção para um setor de moldagem de artefatos plásticos. Vol. 1542, CEUR Workshop Proceedings. Universidade Federal de Juiz de Fora.; 2015.
15. Rocha PL. Um problema de sequenciamento de maquinas paralelas nao relacionadas com tempos de preparação dependentes de máquina e da sequência: Modelos e

Algoritmo exacto. Universidade Federal de Minas Gerais; 2006.

16. Amorim J. A flexible scheduling system for industrial operation. Faculdade de Engenharia da Universidade do Porto; 2009.
17. Pinedo ML. Scheduling: Theory, algorithms, and systems: Fourth edition. Vol. 9781461423, Scheduling: Theory, Algorithms, and Systems: Fourth Edition. 2012. 1-673 p.
18. Conway RW, L. Maxwell W, W. Miller L. Theory of Scheduling. 2003;
19. Beirão N de CL. Sistema de Apoio à Decisão para Sequenciamento de Operações em Ambientes "Job-Shop". Faculdade de Engenharia. Universidade do Porto; 1997.
20. Bou Shaala, A., Shouman, M. A., and Esheem S. Some Heuristic Rules for Job Shop Scheduling Problem. :1–8.
21. Williams HP. Integer programming. In: Constraints. 1980. p. 272–319.
22. Onfroy B, Cohen N. How to speed-up hard problem resolution using GLPK ? 2010;1–19.
23. Santos AM dos. Um modelo de otimização linear inteira, com variáveis binárias, para a resolução do problema sudoku. In São Paulo; 2012.
24. Vaz I, Monteiro T, Fernandes E. Ambientes de Programação e Interface para Problemas de Optimização. Universidade do Minho;
25. Makhorin A. GNU Linear Programming Kit. Vol. 58, Andrew Makhorin. Moscow; 2016.
26. Hillier. F, Lieberman GJ. Introduction to Operations Research.
27. Kawamura MS. Aplicação do método branch-and-bound na programação de tarefas em uma única máquina com data de entrega comum sob penalidades de adiantamento e atraso. Xxvi Enegep. São Paulo; 2006.

# ANEXO A

**Tabela 7-Instância 1**

| Job | Proc. Time (H) | Due Date (horas) | Mould | Desmont | Mont |
|---|---|---|---|---|---|
| **1** | 0,89 | 264 | 1 | 0,66 | 0,75 |
| **2** | 0,18 | 120 | 2 | 0,33 | 0,5 |
| **3** | 20,01 | 288 | 3 | 1 | 0,25 |
| **4** | 4,41 | 264 | 4 | 1 | 0,25 |
| **5** | 10,39 | 144 | 5 | 1 | 0,75 |
| **6** | 5,33 | 168 | 6 | 0,66 | 0,25 |
| **7** | 20,18 | 48 | 6 | 0,66 | 0,25 |
| **8** | 0,48 | 240 | 7 | 1 | 0,75 |
| **9** | 1,93 | 48 | 7 | 1 | 0,75 |
| **10** | 25,35 | 96 | 7 | 1 | 0,75 |
| **11** | 32,38 | 264 | 8 | 1 | 0,5 |
| **12** | 3,22 | 192 | 8 | 1 | 0,5 |
| **13** | 10,47 | 312 | 8 | 1 | 0,5 |
| **14** | 14,03 | 168 | 8 | 1 | 0,5 |
| **15** | 12,29 | 96 | 9 | 0,33 | 0,75 |
| **16** | 13,14 | 312 | 9 | 0,33 | 0,75 |
| **17** | 7,67 | 72 | 9 | 0,33 | 0,75 |
| **18** | 7,78 | 264 | 9 | 0,33 | 0,75 |
| **19** | 27,44 | 168 | 9 | 0,33 | 0,75 |
| **20** | 4,51 | 120 | 10 | 0,66 | 0,75 |
| **21** | 6,52 | 48 | 10 | 0,66 | 0,75 |
| **22** | 25,07 | 192 | 10 | 0,66 | 0,75 |
| **23** | 5,81 | 96 | 10 | 0,66 | 0,75 |
| **24** | 21,37 | 96 | 10 | 0,66 | 0,75 |
| **25** | 12,62 | 288 | 10 | 0,66 | 0,75 |
| **26** | 0,15 | 192 | 11 | 1 | 0,5 |
| **27** | 3,39 | 120 | 11 | 1 | 0,5 |
| **28** | 2,31 | 312 | 11 | 1 | 0,5 |
| **29** | 8,94 | 48 | 11 | 1 | 0,5 |
| **30** | 33,65 | 264 | 11 | 1 | 0,5 |
| **31** | 6,29 | 72 | 11 | 1 | 0,5 |
| **32** | 60,65 | 24 | 11 | 1 | 0,5 |

**Tabela 8-Alocação Moldes/Máquina Instância 1**

| Moldes | Máquina 1 | Máquina 2 |
|---|---|---|
| **1** | x | x |
| **2** | x | x |
| **3** | | x |
| **4** | | x |
| **5** | x | |
| **6** | x | x |
| **7** | x | x |
| **8** | x | x |
| **9** | x | x |
| **10** | | x |
| **11** | | x |

# ANEXO B

**Tabela 9-Instância 2**

| Tarefas | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| **Família** | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 3 | 3 | 3 |
| **Tempo processamento** | 3 | 8 | 9 | 5 | 2 | 6 | 11 | 4 | 10 | 7 |
| **Data de Entrega** | 11 | 18 | 16 | 17 | 27 | 19 | 15 | 12 | 21 | 26 |
| **Tempo de Setup** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Tabela 10-Alocação Moldes/Máquina Instância 2**

| Moldes | Máquina 1 | Máquina 2 |
|---|---|---|
| **1** | **x** | **x** |
| **2** | | **x** |
| **3** | **x** | |