Faculdade de Engenharia da Universidade do Porto

Simulation-Optimization Strategies for the Aerospace Industry

Carlos José Martins de Carvalho

Dissertação realizada no âmbito do

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Major Automação

Orientador: Jorge Manuel Pinho de Sousa

Coorientador: João Pedro Tavares Vieira Basto

Junho de 2020

Resumo

As linhas de produção tradicionais apresentam uma forma atrativa de resolver os problemas de produção em massa. Durante muito tempo estas soluções permitiram cumprir com as necessidades das empresas, onde a variabilidade dos produtos produzidos era muito reduzida. No entanto, com a evolução dos mercados globais, é necessário que estas empresas desenvolvam processos de customização em massa de produtos de uma forma eficiente para que possam manter-se competitivas. Uma boa maneira de tornar estas linhas de produção mais eficientes passa por fazer uma correta alocação dos recursos disponíveis, maximizando a eficiência e evitando os tempos mortos de cada linha.

Esta dissertação estuda um problema de um caso real onde se pretende realizar uma correta alocação de operadores a diferentes tarefas de produção, combinando métodos de otimização com técnicas de simulação. Na sua essência, é tratado um problema de Balanceamento de Linhas de Produção.

A componente de otimização procura alocar cada um dos recursos disponíveis às estações de trabalho existentes enquanto a componente de simulação, além de validar os resultados obtidos na otimização, permite avaliar cada uma das soluções propostas sem interferir no processo produtivo.

Abstract

Traditional production lines present an attractive way of solving mass production problems. For a long time, these solutions made it possible to fulfill the needs of the companies, where the variability of the products produced was very low. However, with the evolution of global markets, it is necessary that these companies develop mass customization product processes in an efficient way so that they can remain competitive. A good way to make these production lines more efficient is to make a correct allocation of available resources maximizing efficiency and avoiding dead times in each line.

This dissertation studies a problem in a real case scenario where it is intended to correctly allocate operators to different production tasks, combining optimization methods with simulation techniques to solve an Assembly Line Balancing problem.

The optimization component tries to allocate each of the available resources to the existing workstations while the simulation component, in addiction to validate the results obtained in the optimization, allows to evaluate each of the proposed solutions without interfering in the production process.

Índice

Lista de Figuras

Lista de tabelas

Abreviaturas e Símbolos

Lista de abreviaturas (ordenadas por ordem alfabética)

GALBP	*Generalized Assembly Line Balancing Problem*
KPI	*Key Performance Indicator*
MALBP	*Mixed-Model Assembly Line Balancing Problem*
MOALBP	*Multi-Objective Assembly Line Balancing Problem*
RALBP	*Robotic Assembly Line Balancing Problem*
SALBP	*Simple Assembly Line Balancing Problem*
UALBP	*U-shaped Assembly Line Balancing Problem*
UML	*Unified Modeling Language*

Capítulo 1 – Introdução

1. Contexto

As linhas de produção tradicionais são atualmente uma forma muito atrativa de solucionar problemas de produção em massa. Já desde os tempos de Henry Ford que existem vários tipos de linhas de produção, quer sejam linhas retas, postos de trabalho em paralelo, modelos multilineares ou linhas em forma de "U"[1].

No entanto, com o avanço da tecnologia e a globalização dos mercados, as empresas cada vez mais alargam a sua carteira de clientes. Se uma empresa quiser ser competitiva terá de prever que os seus clientes podem estar em qualquer parte do mundo e terá de saber lidar com as dificuldades que esta globalização apresenta.

Uma vez que os clientes de uma empresa podem pertencer a qualquer parte do globo, na altura de fazer a conceção dos produtos que vão ser disponibilizados para o mercado, cada empresa deve ter em consideração que o mercado atual se caracteriza por uma variabilidade muito elevada[2]. Além disso, o mercado cada vez apresenta uma concorrência mais diversificada que provoca a necessidade da empresa de praticar preços mais competitivos. É por isso que a existência de um processo de customização em massa de produtos que seja eficiente é vital para as empresas atuais.

As primeiras linhas de produção estavam desenhadas para produzir produtos cuja variabilidade era muito limitada, mas atualmente estas linhas de produção devem ser capazes de analisar os ambientes industriais e adaptar-se de forma fácil e rápida às alterações detetadas. Só desta forma é possível oferecer aos clientes uma gama de produtos com grande variabilidade, a um preço acessível e sem incorrer em custos astronómicos para as empresas.

Uma boa maneira de implementar estas linhas de produção flexíveis passa por fazer um *balanceamento de linhas* otimizado, que permite fazer reestruturações dos recursos de uma empresa para melhorar o processo produtivo sem muitos custos associados. O balanceamento de linhas de produção permite que sejam alocados recursos em diferentes postos de trabalho de forma a que os tempos de processamento desses postos sejam aproximadamente iguais[3].

Finalmente, um exercício de *simulação* permite observar quais serão os resultados esperados das mudanças efetuadas mediante o balanceamento das linhas de produção, de forma a saber se os níveis de performance pretendidos são alcançados. Com a simulação também podemos analisar cenários que ficam de fora do âmbito da otimização, como por exemplo a estocasticidade dos tempos de processamento, avarias não programadas ou fazer o estudo da alocação de recursos de forma dinâmica, permitindo deslocamentos entre postos de trabalho quando alguma estação se encontra livre. Com os resultados da simulação podem ser tomadas decisões informadas sobre aspetos a implementar na linha de produção.

2. Motivação

O trabalho realizado utiliza os dados de um caso real. Nesta ocasião é estudado um problema presente numa fábrica da empresa Embraer, que produz asas de aviões. Esta fábrica possui três linhas de produção na planta fabril. O principal desafio deste trabalho passa por alocar os trabalhadores da fábrica nos diferentes postos de trabalho de forma a fazer um correto balanceamento das linhas de produção utilizando um *software* de *otimização* apropriado para o efeito.

Como foi referido anteriormente um bom balanceamento das linhas de produção permite distribuir os recursos pelos diferentes postos de trabalho de forma a que os tempos de processamento de cada posto sejam aproximadamente iguais. Com isto, é possível diminuir o tempo de ciclo da linha e com isso aumentar a produtividade.

No caso em estudo é particularmente difícil quantificar os indicadores de desempenho da fábrica, nomeadamente *lead times* e as capacidades de produção de cada posto de trabalho. Uma vez que essa informação não está bem definida, é muito complicado para a empresa saber que decisões tomar de forma a melhorar a produtividade da fábrica.

A utilização de um *software* de *simulação* irá permitir ter em atenção um número muito maior de variáveis que os modelos de otimização tradicionais não consideram, como por exemplo fatores relativos à estocasticidade dos tempos de processamento de cada posto. Também podem ser consideradas situações de avarias de máquinas e situações de alocações dinâmicas de recursos.

Uma vez que todas as soluções que possam vir a ser encontradas podem ser testadas no software de simulação, podemos avaliar quais os resultados que esperamos obter sem ter de alterar o *layout* da planta de produção nem ter de parar o processo produtivo. Juntando o facto de o balanceamento de linhas de produção não interferir na quantidade de recursos da empresa, mas sim na organização destes recursos, podemos dizer que o método de simulação permite avaliar as soluções sem obrigar a empresa a ensaiar na prática essas soluções, evitando assim custos que podem ser significativos.

Uma vez que os problemas de balanceamento de linhas de produção são bastante comuns nas indústrias atuais, os resultados que venham a ser apresentados nesta dissertação podem ser estendidos, com as devidas adaptações, para outros ambientes de produção.

3. Objetivos

Em primeiro lugar é criado um cenário de *simulação* que represente o estado atual da planta da fábrica, representando as diferentes linhas de produção, os tempos de produção em cada estação de trabalho e os recursos alocados em cada posto. Desta forma poderemos ter uma visão geral do estado atual da fábrica.

De seguida é feito um estudo de várias opções de *balanceamento das linhas* de produção. Este estudo deverá ter em conta as características de cada linha de produção: número de operadores, tempos de processamento, regras de precedências nas estações de trabalho e

outros fatores limitativos. No final será possível obter uma série de ações a implementar nas linhas de produção que se espera melhorar a performance da fábrica.

Posteriormente é realizado um trabalho de simulação de modo a validar se essas ações definidas anteriormente são efetivamente positivas ou negativas para o desempenho da fábrica.

No final, após recolher todas as medidas que foram testadas e que geraram melhorias no processo produtivo, são analisadas as ações propostas de modo a obter sugestões para melhorar os KPI's do sistema de produção.

Capítulo 2 – Revisão Bibliográfica

Neste capítulo vão ser apresentados diferentes conceitos importantes para o posterior desenvolvimento da dissertação. Todos as informações apresentadas neste capítulo sintetizam um trabalho de pesquisa feito em diferentes publicações, as quais se encontram referenciadas no final do documento.

Na secção 1 será feita uma introdução às linhas de produção e serão apresentados diferentes conceitos relacionados com o balanceamento destas linhas de produção.

Posteriormente, na secção 2, irão ser apresentados os diferentes tipos de problemas de balanceamento que são abordados na literatura, quais as suas características principais e alguns modelos que já existem para sistematizar alguns destes problemas.

Na secção 3 será feita uma introdução aos métodos de otimização com referência tanto a métodos exatos assim como a métodos aproximados.

Finalmente, na secção 4, vão ser apresentados conceitos base de simulação e vai ser feita uma introdução à importância que este tipo de técnicas apresenta para as diferentes indústrias atuais.

1. Linhas de produção

Num determinado ambiente fabril podem existir diferentes tipos de *layout* em função das várias características do produto que vai ser trabalhado (volume de produção, processo, dimensões do produto). No nosso caso iremos abordar um tipo de *layout* linear (linha de produção) que é aplicado em situações onde o processo de transformação implica várias operações individuais (tarefas) que podem ser realizadas em várias máquinas ou postos de trabalho.

Cada tarefa do processo produtivo só pode ser realizada se cumprir com as respetivas restrições de precedência, e tem um tempo de processamento associado. As restrições de precedência irão determinar a ordem de cada uma das tarefas e o tempo de processamento indica o tempo que o produto deve passar em cada tarefa até estar concluído. Todas estas restrições são normalmente apresentadas num diagrama de precedência que permite visualizar todas as operações e relações de uma maneira simples.

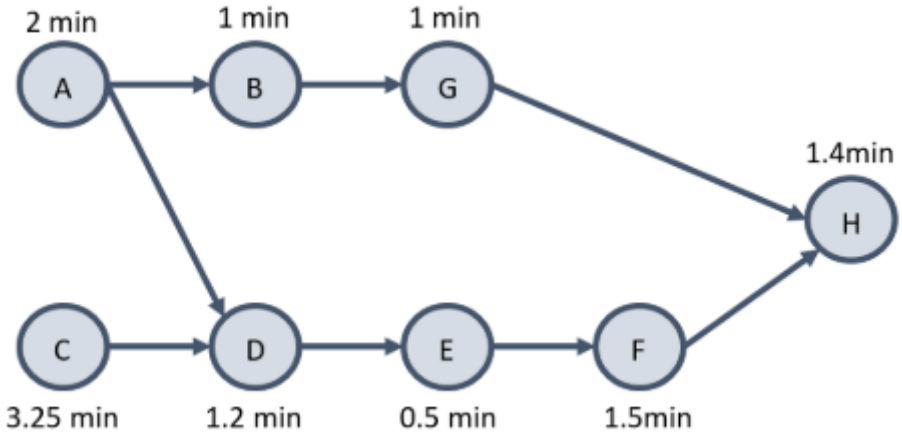

Figura 1 – Diagrama de precedência

Por último, cada uma das estações de trabalho encontram-se interligadas por algum meio de transporte, de forma a que os produtos possam fluir ao longo da linha de produção [4].

Uma linha de produção está então constituída por uma série de estações de trabalho, cuja posição é fixa e cuja sequência é ditada pela lógica das sucessivas operações a realizar[3].

1.1 Conceitos de balanceamento de linhas de produção

Nesta subsecção vão ser apresentados alguns conceitos base que permitem acompanhar melhor o desenvolvimento da dissertação [3].

- **Tempo total de produção** de um produto corresponde ao somatório dos tempos de processamento de todas as operações associadas a esse produto.

$$t_{TOTAL} = \sum_{i}^{k} t_i \tag{1}$$

- **Tempo de ciclo**, que consiste no tempo entre a saída de peças consecutivas, corresponde ainda ao máximo tempo necessário em cada estação de trabalho.

$$Tempo\ de\ ciclo = c = \frac{Tempo\ disponível\ por\ periodo}{Número\ de\ peças\ pretendidas\ por\ periodo} \tag{2}$$

- **Número mínimo de estações de trabalho necessárias**, calculado a partir da seguinte equação, deve ser arredondado sempre para o valor inteiro imediatamente superior.

$$n_{min} = \frac{\sum_{i=1}^{k} t_i}{c} \tag{3}$$

- **Eficiência** da solução adotada.

$$e = \frac{\sum_{i=1}^{j} t_i}{c} - n\ x\ c \tag{4}$$

- **Tempo total de paragem**, ou folga de uma linha de produção.

$$IT = n\ x\ c - \sum_{i=1}^{k} t_i \tag{5}$$

Onde:

n = Número de estações de trabalho;

c = Tempo de ciclo da linha de produção;

t_i = Tempo necessário para completar a tarefa i da linha de produção;

k = Número total de operações a serem executadas na linha de produção.

2. Balanceamento de linhas de produção

Uma vez que estão definidas todas as tarefas que são necessárias realizar para completar o produto, estas tarefas são agrupadas em estações de trabalho. Por norma, uma estação de trabalho irá realizar várias tarefas que estejam relacionadas, sem violar as restrições de precedência impostas no processo produtivo. Como cada tarefa apresenta um tempo de processamento próprio, o problema de balanceamento de linhas de produção reside na alocação de tarefas a postos de trabalho de forma a que todos os postos tenham aproximadamente a mesma quantidade de trabalho atribuída [5]. Por outras palavras, cada estação de trabalho deve ter, na totalidade das tarefas que lhe são alocadas, o mesmo tempo de processamento.

O principal objetivo do balanceamento consiste em permitir (na medida do possível) um fluxo de materiais rápido e uniforme através da linha de produção e, ao mesmo tempo, obter a máxima utilização do potencial humano, máxima utilização de maquinas, equipamentos e ferramentas e um dispêndio mínimo de material ou tempo de processo [3].

Com isto podem ser obtidos os seguintes resultados:

- Minimização das estações de trabalho
- Minimização do tempo de ciclo

2.1 Balanceamento de linhas

Segundo o tipo de produto para o qual a linha de produção vai ser dimensionada existem dois tipos de modelos que estudam o problema do balanceamento da linha de produção [6]:

- Modelo simples: SALBP (*Simple Assembly Line Balancing Problem*)

- Modelo generalizado: GALBP (*Generalized Assembly Line Balancing Problem*)

2.1.1 Modelo SALBP

O modelo SALBP é orientado à produção de um único produto numa linha de produção em série com estações de trabalho a um lado da linha de produção [7].

De forma a poder ser utilizado o modelo SALBP, o agrupamento de tarefas a postos de trabalho é feito assegurando que são cumpridas as seguintes condições [8], [9]:

- uma tarefa não pode ser dividida, tendo de ser realizada por completo em apenas um único posto de trabalho;
- a sequência de tarefas afetadas tem de respeitar as restrições de precedência;
- todos os postos de trabalho têm condições para realizar qualquer tarefa;
- os tempos de processamento das tarefas são conhecidos e independentes do posto de trabalho em que se realizam;
- o somatório dos tempos de processamento das tarefas afetadas a cada posto de trabalho não pode exceder o tempo de ciclo;
- todos os parâmetros de entrada são conhecidos com certeza;
- todas as tarefas devem ser completadas;
- o transporte do produto entre estações de trabalho ocorre imediatamente;
- a linha de produção é orientada a um único produto.

Em função dos diferentes objetivos que queremos maximizar, ou minimizar, e os dados que conseguimos obter da linha de produção podemos aplicar um de quatro tipos possíveis de modelos SALBP:

Tabela 1 - Classificação de modelos SALBP

Tipo	Dado	Objetivo
SALBP-1	Tempo de ciclo	Minimizar nº estações de trabalho
SALBP-2	Nº estações de trabalho	Minimizar tempo de ciclo
SALBP-E	-	Maximizar eficiência da linha de produção
SALBP-F	Tempo de ciclo e nº de estações de trabalho	Obter uma solução admissível

De seguida são apresentados os modelos SALBP–1 e SALBP-2.

O modelo SALBP–1 pretende minimizar o número de estações de trabalho numa certa linha de produção sabendo o tempo de ciclo. Este tipo de problemas pode ser modelado utilizando as seguintes variáveis [10]:

i – Número de tarefas. $i = 1, 2, \ldots, n$

j – Número máximo de estações de trabalho. $j = 1, 2, \ldots, k$

c – Tempo de ciclo do produto

t_i – Tempo de operação da tarefa i

F_i – Conjunto de tarefas que só podem ser executadas depois da tarefa *i* estar concluída

e as seguintes variáveis de decisão:

X_{ij} – Variável binária: 1 se à tarefa *i* for executada na estação de trabalho *j*, 0 se não

d_j – Coeficientes atribuídos de forma crescente – $(d_1 << d_2 << \ldots << d_n)$

A função objetivo procura minimizar o número total de estações de trabalho:

$$FO = min \sum_{j=1}^{k} \quad \sum_{i=1}^{n} \quad d_i x_{ij} \tag{6}$$

sujeito às seguintes restrições:

1. Cada tarefa deve ser realizada numa estação de trabalho:

$$\sum_{j=1}^{k} x_{ij} = 1, \forall_i \in I \tag{7}$$

2. Precedências entre as tarefas a realizar:

$$\sum_{j=1}^{J} x_{aj} - \sum_{j=1}^{J} x_{bj} \leq 0, \forall_a \in N, \forall_b \in F_a \tag{8}$$

3. O tempo de ciclo não é excedido:

$$\sum_{j=1}^{J} t_i \, x_{ij} \leq c \, , \forall_j \in J \tag{9}$$

O modelo SALBP–2 pretende minimizar o tempo de ciclo duma certa linha de produção sabendo o número de estações de trabalho existentes nessa linha. Este tipo de problemas pode ser modelado utilizando as seguintes variáveis [9]:

I – Número de tarefas

J – Número de estações de trabalho

P_i – Conjunto de tarefas que devem ser realizadas antes ou ao mesmo tempo que a tarefa *i*

t_i – Tempo necessário para realizar a tarefa *i*

e as seguintes variáveis de decisão:

c – Tempo de ciclo do produto

x_{ij} – Variável binária: 1 se a tarefa *i* for alocada à estação *j*, 0 se não

A função objetivo pretende minimizar o tempo de ciclo:

$$FO = Min\ c \tag{10}$$

sujeito às seguintes restrições:

1. Cada tarefa é atribuída a uma estação de trabalho:

$$\sum_{j=1}^{J} x_{ij} = 1\ \forall_i \in \{1, 2, \dots, I\} \tag{11}$$

2. O tempo de ciclo é tão grande como o tempo necessário para a estação de trabalho com maior duração:

$$\sum_{i=1}^{I} t_i\, x_{ij}\ \leq c\ \ \forall_j \in \{1, 2, \dots, J\} \tag{12}$$

3. A todas as tarefas anteriores a *i* é atribuída uma estação de trabalho antes da estação de *i*, ou a mesma:

$$x_{ik} \leq \sum_{j=1}^{k} x_{hj}\ \ \forall_i \in \{1, 2, \dots, I\},\ \forall_k \in \{1, 2, \dots, J\},\ \forall_h \in\ P_i \tag{13}$$

4. Cada tarefa é realizada em uma única estação de trabalho:

$$x_{ij} = 0{,}1\ \ \forall_i \in \{1, 2, \dots, I\},\ \ j \in \{1, 2, \dots, J\} \tag{14}$$

2.1.2 Modelo GALBP

O modelo GALBP é uma generalização do modelo SALBP, onde uma ou varias hipóteses do caso mais simples são alteradas [11]. Dependendo de quais forem as hipóteses alteradas iremos distinguir diferentes tipos de problemas:

- UALBP: Envolve linhas de produção em forma de "U".
- MALBP: Considera uma linha de produção com mais de um produto, ou diferentes modelos de um mesmo produto.
- MOALBP: Considera vários objetivos de otimização.
- RALBP: Utilizado para linhas robóticas.

Ainda assim, na literatura, é possível encontrar outros tipos de problemas menos comuns que envolvem estações de trabalho paralelas ou tarefas em paralelo [12], estações de trabalho com diferentes tipos de equipamento que vai obrigar a efetuar uma escolha do equipamento a ser utilizado [13], linhas de produção com *buffers* ou vários problemas relacionados com tempos de processamento sequenciais, estocásticos ou difusos [11].

Atualmente existem diferentes métodos de resolução de problemas relacionados com balanceamento de linhas de produção, mas podemos considerar que todos estes métodos se encaixam em um de dois grupos: métodos exatos ou métodos aproximados [11].

A maioria dos métodos exatos estão baseados em técnicas de programação dinâmica e procedimentos de "*Branch and Bound*". Por outro lado, uma grande variedade de métodos aproximados têm servido para resolver problemas de balanceamento de linhas de produção de uma forma mais realista, entre os quais podemos encontrar heurísticas baseadas em regras de prioridade e processos de enumeração, meta-heurísticas, como algoritmos genéticos, "*simulated annealing*" ou procura tabu. É feita uma análise mais aprofundada de alguns destes métodos em [14], [15] e [16].

2.1.3 *Resource-Constrained Project Scheduling*

O caso de estudo aborda um problema de balanceamento que devido à natureza das operações a serem realizadas necessita de uma restrição adicional: para um dado conjunto de tarefas a realizar numa estação e com um determinado número de operadores alocado em cada estação, devemos encontrar uma forma de definir que tarefas são realizadas por que operador e em que ordem de forma a minimizar o tempo de ciclo de cada estação de trabalho.

Este tipo de problema é próximo do problema de "*Resource-Constrained Project Scheduling*" [17] e já foi abordado pela literatura no âmbito da calendarização de projetos.

Neste tipo de problemas o trabalho a realizar é composto por um conjunto de atividades J, denominadas tarefas, rotuladas $j = 1, 2, \ldots, J$. Devido a requisitos tecnológicos existem relações de precedência entre algumas das tarefas definidas. Estas relações de precedência são dadas como grupos de precedências, P_j, que indicam que a tarefa j só pode ser realizada assim que todas as suas tarefas precedentes tenham sido completadas. De forma análoga existe um grupo de atividades sucessoras, S_j, que correspondem às tarefas que podem ser realizadas depois da tarefa j ter sido concluída.

À exceção da atividade inicial ($j = 0$) e a atividade final ($j = J + 1$) cada uma das tarefas a realizar necessitam de uma quantidade de recursos para serem realizadas. O conjunto dos recursos disponíveis é definido por K e a disponibilidade de cada recurso $k \in K$ por período é constante e dada por R_k.

O tempo de processamento de cada atividade j é representado por p_j e requer os recursos k por um período r_{jk}. Além disso, sempre que uma tarefa é iniciada não pode ser interrompida.

É assumido que as atividades inicial e final têm uma duração de zero períodos e não requerem nenhum recurso.

O objetivo deste tipo de problemas consiste em obter uma lista ordenada (calendário) que apresente o menor tempo de ciclo possível respeitando as restrições de precedências e de recursos definidos.

Na figura 2 é possível observar um exemplo deste tipo de problemas.

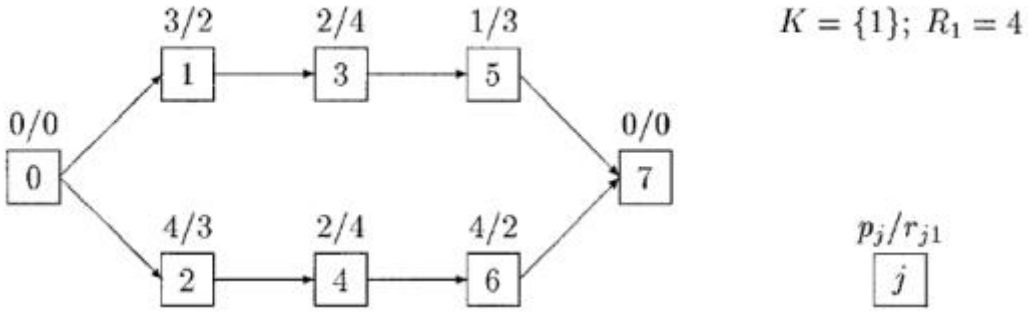

Figura 2 – Exemplo de aplicação de um problema "*Resource-Constrained Project Scheduling*"[17]

3. Métodos de Otimização

Os problemas de balanceamento de linhas de produção são, na sua essência, problemas de otimização. Justifica-se por isso fazer um estudo de quais são os diferentes métodos de otimização e quais as suas vantagens e desvantagens.

A otimização busca responder à pergunta "O que é melhor?" para problemas em que a qualidade de uma resposta pode ser medida por um número [18]. Para conseguir obter o resultado ótimo a maioria dos problemas de otimização procuram a melhor configuração de variáveis de forma a atingir um determinado objetivo mesurável [19].

Uma abordagem matemática implica encontrar um valor de $s^* \in S$ tal que:

$$f(s^*) \leq f(s)\ (\ ou\ f(s^*)\ \geq f(s)\ )\ \forall s\ \in\ S \tag{15}$$

Estes tipos de problemas de otimização aparecem divididos em dois grandes grupos, os que apresentam apenas variáveis reais e os que apresentam também variáveis discretas. Iremos aprofundar mais sobre os problemas que incluem variáveis discretas uma vez que são este tipo de problemas os que vamos encontrar na dissertação.

Existem duas abordagens possíveis para aplicar métodos de otimização a problemas com variáveis discretas: métodos exatos e métodos aproximados.

3.1 Métodos exatos

Os métodos exatos garantem encontrar, em instâncias de tamanho finito, a solução ótima à custa de um tempo computacional que cresce exponencialmente com a dimensão do problema [19]. Isto significa que podem existir situações onde não seja possível apresentar a solução ótima de um problema porque o tempo necessário para encontrar essa solução, mesmo que exista, é inviável.

A maioria dos métodos exatos baseiam-se em soluções de programação dinâmica e procedimentos *"Branch and Bound"*.

3.2 Métodos aproximados

Os métodos aproximados, ao contrário dos métodos exatos, não garantem a solução ótima do problema, mas apresentando técnicas mais simples, rápidas e flexíveis, permitem obter soluções de boa qualidade em tempos aceitáveis. Podemos dizer então que a diferença para os métodos aproximados reside no compromisso implementado entre eficácia (obter uma solução o mais próxima possível da solução ótima) e eficiência (obter uma boa solução num determinado tempo).

Dentro dos métodos aproximados podemos considerar três abordagens principais: métodos construtivos, métodos de pesquisa local e meta-heurísticas [19]. Os métodos construtivos são utilizados para procurar soluções iniciais. Normalmente são rápidos a encontrar soluções, mesmo que estas não sejam de boa qualidade. Os métodos de pesquisa local procuram melhorar as soluções iniciais fazendo pesquisas nas vizinhanças. As meta-heurísticas são métodos que conjugam pesquisas locais e estratégias de mais alto nível de forma a criar processos que permitam fugir a soluções ótimas locais [20].

4. Simulação de Eventos Discretos

A simulação baseada em eventos discretos representa a modelação, simulação e análise de sistemas utilizando técnicas matemáticas e computacionais. Estas técnicas permitem criar um modelo conceptual da estrutura que compõe o sistema e o seu comportamento [21]. Devido à capacidade de representar quantitativamente ambientes reais, simulando as dinâmicas de um determinado sistema, este tipo de método de simulação revela-se uma poderosa ferramenta de análise capaz de avaliar fluxos produtivos numa empresa, desde atividades singulares ao impacto das mesmas na globalidade do processo [22].

São apresentados de seguida um conjunto de conceitos relacionados com a simulação baseada em eventos discretos que permitem perceber melhor o funcionamento deste tipo de ferramentas [22]:

- *Entidade*: qualquer componente do sistema que requer uma representação explícita no modelo (máquinas, clientes…).
- *Sistema*: coleção de entidades que interagem entre si para atingirem um determinado objetivo.
- *Modelo*: representação abstrata do sistema, contendo relações lógicas, estruturais ou matemáticas capazes de descrever o sistema em termos de estado, atributos, processos ou atividades.
- *Estado do Sistema*: representa o conjunto de variáveis capazes de descrever o sistema num determinado instante de tempo.
- *Atributos*: propriedades de uma entidade.
- *Evento*: ocorrência instantânea que muda o estado do sistema.
- *Atividade*: duração de tempo específica em que é conhecido o seu momento de início.

Num modelo de simulação baseado em eventos discretos uma série de variáveis de estado definem um conjunto de atividades a serem desenvolvidas, num momento temporal discreto. Consoante vão sendo verificadas ocorrências de eventos, novos estados do sistema são

alcançados e, por consequente, novas atividades são desenvolvidas. No sistema, as várias entidades competem por diferentes recursos, ou aguardam a que estes recursos sejam liberados por outras entidades.

Dado a grande flexibilidade que este tipo de ambientes de simulação proporciona, cada vez mais empresas estão dispostas a investir nesta tecnologia, de forma a poder tornar os seus processos produtivos mais flexíveis. Fazendo uma análise metódica do modelo de simulação implementado, é possível concluir que partes do processo produtivo apresentam falhas e estudar possíveis soluções de forma a mitigar o problema. A avaliação de várias soluções também é feita no modelo de simulação uma vez que permite quantificar o grau de melhoria que cada solução apresenta, sem a necessidade de despender recursos na aquisição do material [22].

Capítulo 3 – Caracterização do problema

Neste capítulo será apresentada uma caracterização detalhada do problema que vai ser tratado durante a dissertação assim como a sequência de trabalhos proposta para obter uma solução satisfatória.

1. Definição do problema

1.1 Caso de estudo

O caso de estudo a ser desenvolvido trata um problema numa empresa da indústria aeronáutica. Neste caso, a construtora brasileira Embraer possui, em Évora (Portugal), uma fábrica encarregada de fazer a produção das asas dos aviões. Atualmente esta construtora ocupa a terceira posição mundial no setor e é responsável pela construção de diversos tipos de aviões: comerciais, executivos, agrícolas e ainda militares.

A fábrica que vai ser estudada apresenta duas linhas de produção distintas (ver figura 3): a linha 1, representada pela cor amarela, e a linha 2, representada pela cor laranja. A linha de produção 1 está, por sua vez, composta por duas linhas distintas. Sendo assim, a linha 1 é composta pelas linhas 1.1 e 1.2. Cada linha de produção produz um produto diferente, logo podemos concluir que a fábrica é responsável pela construção de três produtos diferentes.

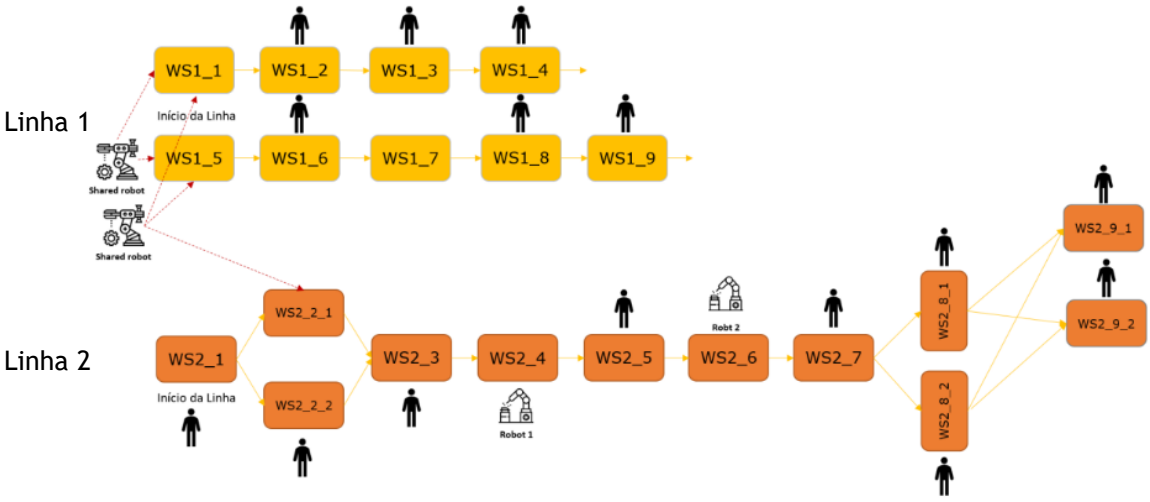

Figura 3 - Layout da fábrica

A fábrica possui três tipos de recursos para realizar cada uma das tarefas nas estações de trabalho correspondentes: robô partilhado, robô fixo e operadores humanos.

Robô partilhado

Na fábrica existem dois robôs partilhados localizados no início das linhas de produção. Estes robôs são responsáveis por várias tarefas a realizar nas estações WS1_1, WS1_5 e WS2_2_1, e movimentam-se de acordo com a ordem dos pedidos que recebem das estações de trabalho. Quando não se encontram a trabalhar, aguardam num local designado *HomeNode*. No âmbito

desta dissertação é considerado que todas as tarefas realizadas pelos robôs partilhados são feitas automaticamente, isto é, assume-se um número infinito de robôs partilhados disponíveis.

Robô fixo

Na fábrica só existem dois recursos deste tipo. São responsáveis por realizar as tarefas da estação de trabalho onde estão alocados uma vez que não se podem deslocar. Estes recursos encontram-se nas estações WS2_4 e WS2_6.

Operadores humanos

Cada operador é alocado a uma estação de trabalho fixa. Dependendo da estação de trabalho em questão, existe um número máximo de operadores que podem estar alocados a essa estação. Esse número máximo de trabalhadores aparece descrito na tabela 2:

Tabela 2 - Número máximo de operadores por estação de trabalho

ESTAÇÃO	Nº MÁXIMO DE OPERADORES
WS1_1	4
WS1_2	4
WS1_3	4
WS1_4	4
WS1_5	12
WS1_6	6
WS1_8	6
WS1_9	6
WS2_1	4
WS2_2_1	4
WS2_2_2	4
WS2_3	7
WS2_5	7
WS2_7	7
WS2_8_1	10
WS2_8_2	10
WS2_9_1	3
WS2_9_2	3

Por último é necessário prestar atenção a duas situações especiais:

Em primeiro lugar, na linha 2, existe uma divisão de trabalho. Isto é, dois tipos de componentes diferentes chegam à estação WS2_1. Posteriormente um destes componentes será direcionado para a estação WS2_2_1 enquanto que o outro irá ser trabalhado na estação WS2_2_2. Só na estação WS2_3 é que ambos os componentes são combinados, percorrendo então o resto da linha de produção na forma de um único componente.

Mais à frente, os pares de estações WS2_8_1/WS2_8_2 e WS2_9_1/WS2_9_2 realizam o mesmo tipo de tarefas. Isto significa que podem existir situações onde não seja vantajoso estarem as duas estações a trabalhar em paralelo, disponibilizando operadores para serem alocados a outras estações de forma a agilizar o processo produtivo e diminuir o tempo de processamento.

1.2 Objetivo da dissertação

A definição e obtenção dos KPIs de cada linha de produção é um processo bastante complicado e, muitas vezes, a falta desta informação dificulta os momentos de tomadas de decisão em momentos críticos da gestão estratégica das empresas. Posto isto, um dos problemas que vai ser abordado passa pela correta obtenção dos KPIs do processo produtivo.

Outro problema apresentado neste ambiente fabril reside na alocação dos recursos disponíveis. Uma melhor distribuição dos recursos da empresa irá permitir uma diminuição de tempos de espera e tempos "mortos", que afetam diretamente na produtividade.

Fazendo uma junção da informação obtida através da obtenção dos KPIs de cada linha de produção e uma correta alocação dos recursos da empresa, podemos esperar uma melhoria significativa no processo produtivo.

Assim, esta dissertação tem como principais objetivos definir os KPIs associados ao processo produtivo em análise e fazer uma alocação dos recursos disponíveis de forma a aumentar a produtividade das linhas de produção.

2. Abordagem

A abordagem proposta para esta dissertação consta de 4 fases principais:

1. **Criação do modelo de simulação**

Nesta fase irá ser criado um modelo de simulação que represente as linhas de produção presentes na fábrica, incluindo todas as entidades do sistema que pertencem ao problema. O modelo a ser criado deverá ser modular, permitindo dividir cada uma das etapas lógicas do processo em pequenos processos que atuam independentes dos outros, mas que interagem entre si. No modelo de simulação também será necessário criar uma estrutura que permita fazer uma análise uma vez terminado o processo de simulação. Para isto será criado um painel que inclua os KPIs mais relevantes do processo, de forma a poder ser feita uma análise após cada simulação.

2. **Criação do modelo de otimização**

Nesta fase vai ser desenvolvido um modelo matemático que permita calcular a melhor opção de alocação de recursos para cada linha de produção. O modelo matemático deverá ter em conta todas as restrições de precedência impostas pela ordem das tarefas em cada máquina assim como as restrições referentes à alocação dos recursos humanos.

3. **Implementação do modelo de otimização em dois softwares diferentes**

Para realizar um estudo comparativo, o modelo de otimização desenvolvido será implementado em dois softwares de simulação diferentes: "CPLEX" e "OR-Tools". Para isso irá ser desenvolvido um programa utilizando a linguagem de programação *Python* que permita implementar o modelo desenvolvido na etapa anterior. O resultado será a alocação dos operadores selecionados a cada estação de trabalho assim como a ordem das tarefas a realizar. Este resultado deve poder ser interpretado pelo modelo de simulação de forma a poder validar e comparar os resultados obtidos.

4. Análise e comparação dos resultados

Vão ser analisados dois cenários de teste diferentes. No primeiro o objetivo será obter o menor tempo de ciclo possível, assumindo que não existem restrições de operadores nas estações. No segundo o objetivo será finalizar o plano de produção num tempo definido.

Com os resultados obtidos em cada um dos cenários será feita uma comparação para determinar qual das duas ferramentas apresenta um melhor desempenho, assim como as diferenças entre elas. Para efetuar esta comparação, o tempo de otimização disponível para cada ferramenta deverá ser o mesmo, e as soluções obtidas deverão ser colocadas no mesmo modelo de simulação.

Cada uma das fases apresentadas na solução proposta serão detalhadas de seguida.

Capítulo 4 – Simulação

Neste capítulo vai ser apresentado de forma detalhada, o modelo de simulação que foi desenvolvido. Na primeira secção será apresentado o modelo preliminar que foi desenvolvido onde apenas algumas funções básicas da simulação foram adicionadas. Na segunda secção será apresentado o modelo final que engloba todas as funções do modelo de simulação. Por último na terceira secção serão apresentadas as diferentes páginas de *dashboard* que foram criadas.

A ferramenta escolhida para criar o modelo de simulação do problema em estudo foi o *software* de simulação *Flexsim*. A principal razão que levou à escolha de esta ferramenta reside na facilidade de implementação de estruturas complexas de simulação. Devido à modelação de sistemas utilizando a ferramenta *Process Flow* incluída é possível dividir o problema em atividades unitárias, que ligadas entre si geram blocos de atividades que realizam funções específicas do modelo. Estes blocos de atividades também podem ser ligados a outras atividades individuais ou grupos de atividades. Ao mesmo tempo também é possível implementar páginas que recolhem informação da simulação e a apresentam de forma simples. Outra característica importante é a existência de uma ferramenta de *debug* que permite identificar e corrigir erros que ocorrem no desenvolvimento do modelo de simulação.

Os elementos de ligação entre o modelo 3D e o *Process Flow* denominam-se *tokens*. Cada *token* esta associado a um elemento do modelo 3D, e percorre as diversas atividades do *Process Flow* realizando cada uma das ações definidas no modelo.

1. Modelo preliminar

Na primeira fase de trabalho foi criado um modelo onde foram posicionadas todas as máquinas na planta de trabalho e foi considerado que cada máquina apenas realizava uma tarefa. Neste primeiro modelo já foi possível implementar a possibilidade de serem produzidos 3 tipos de peças, como está definido no problema em análise. As regras de precedências entre cada máquina foram respeitadas. Ao contrário do que tinha sido descrito na definição do problema, todas as peças definidas para produção são armazenadas na estação *ProductionQueue*, e depois são colocadas automaticamente na máquina que as vai processar, aguardando unicamente que tal máquina se encontre disponível para receber o produto.

Podemos observar na figura 4 o aspeto gráfico da simulação e na figura 5 o *Process Flow* que controla o modelo preliminar.

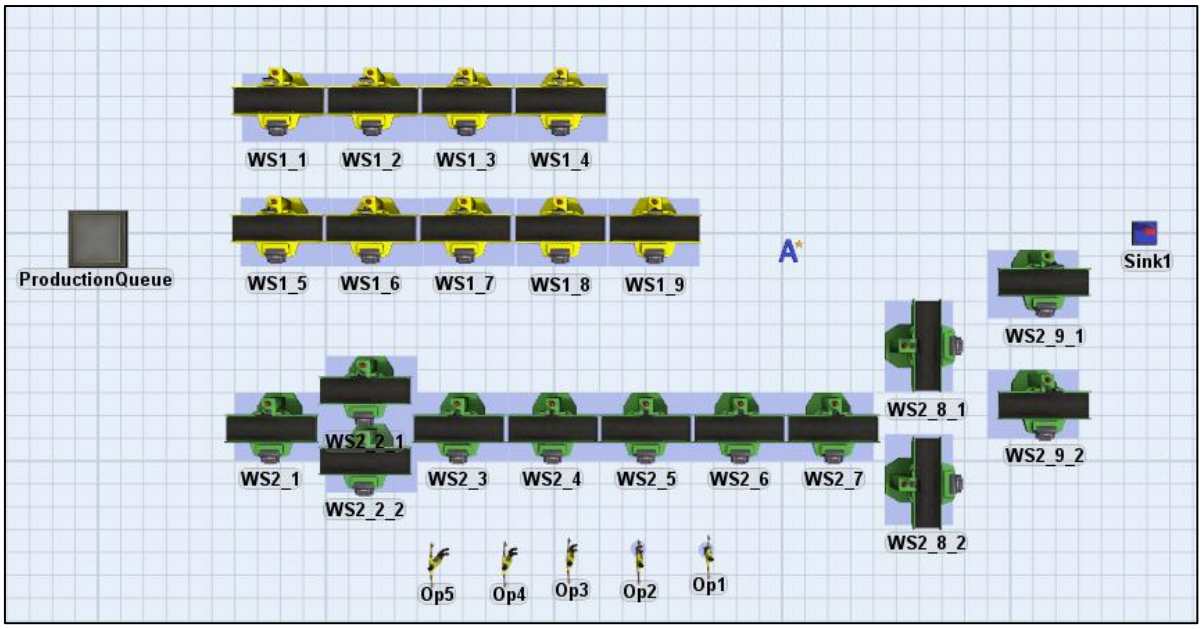

Figura 4 - Aspeto gráfico do modelo de simulação

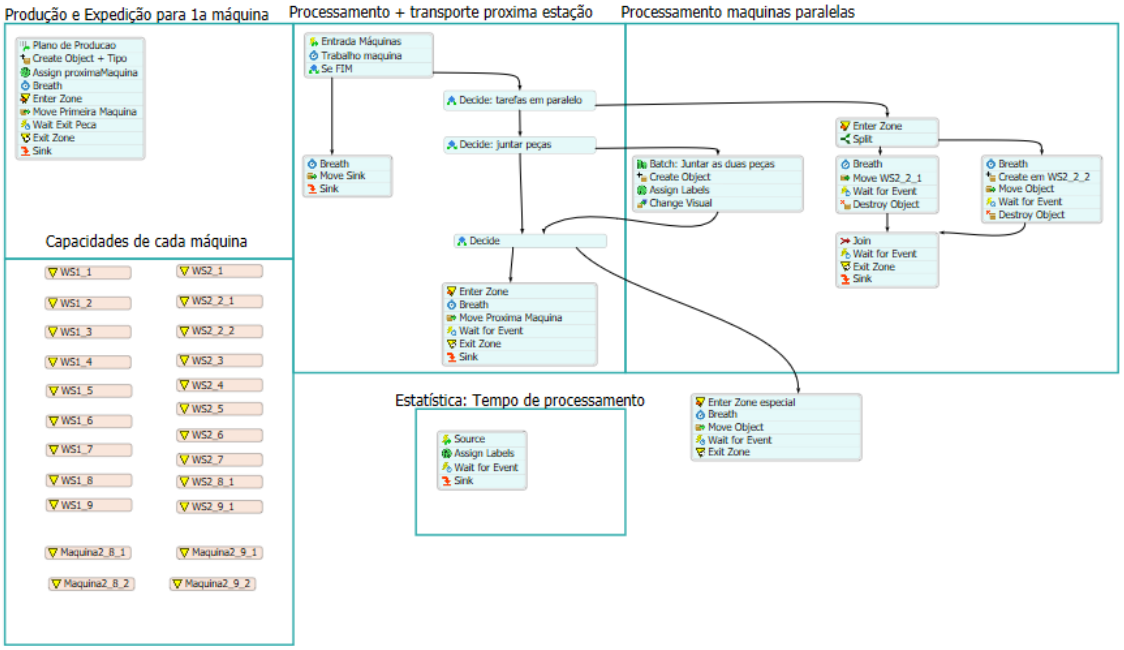

Figura 5 - *Process Flow* do modelo preliminar

Podemos observar que o *Process Flow* está dividido em vários fluxogramas, formados por vários grupos de atividades, onde cada um deles cumpre uma função específica no modelo de simulação. A função e funcionamento de cada fluxograma será apresentado de seguida:

1.1 Produção e expedição para a 1ª máquina

O fluxograma da figura 6 apenas contém um grupo de atividades e tem como função principal criar as peças definidas no plano de produção e colocá-las na respetiva máquina que as vai processar.

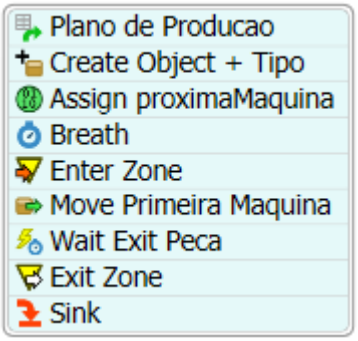

Figura 6 – *Process Flow*, 'Produção e expedição para a 1º máquina'

Em primeiro lugar a atividade 'Plano de Producao' recebe de uma folha Excel como a da figura 7 a quantidade de peças de cada tipo que devem ser criadas assim como o tempo em que cada peça vai ser criada ('Release Date'). Por cada peça do plano de produção é criado um *token*, que ao passar a atividade 'Create Object + Tipo' cria no modelo 3D um objeto gráfico que representa essa peça. Todas as peças são criadas na 'ProductionQueue'. Na atividade 'Assign proximaMaquina' é criada uma variável ligada a cada *token* que indica qual a próxima máquina para onde o objeto associado a esse *token* se vai dirigir. O modelo sabe indicar a próxima máquina porque vai seguindo a ordem das máquinas definidas na tabela 'tarefas' (tabela 3).

De forma a garantir que uma peça pode ser transportada para a próxima máquina, devemos garantir que a máquina de destino se encontra livre para receber tal peça. A forma como foi solucionado este problema passa pela implementação de zonas no '*Process Flow*' do modelo de simulação.

Uma zona é uma atividade do '*Process Flow*' que limita o número de *tokens* que podem permanecer dentro dessa atividade. Para introduzir um *token* numa zona utilizamos a atividade '*Enter Zone*' e para retirar um *token* utilizamos '*Exit Zone*'. Se uma zona estiver ocupada e algum *token* quiser entrar, este aguardará na atividade '*Enter Zone*' até que seja possível entrar na zona.

No nosso modelo de simulação criamos uma zona para cada máquina, de forma a poder controlar quantas peças se encontram em cada momento em cada máquina. Cada zona apenas aceita um *token* de cada vez, limitando a capacidade das máquinas a 1 peça.

Para que a peça que foi criada possa avançar para a primeira máquina, tem primeiro de entrar na zona da máquina correspondente, passando na atividade 'Entra Zona 1ª maquina'. De seguida a peça é transportada para a máquina correspondente na atividade 'Mover Primeira Maquina' e o *token* aguarda que a peça saia da primeira máquina na atividade 'Espera saída Peca'. Quando a peça que estava a ser processada segue para a seguinte máquina da linha de produção, o *token* avança para a atividade 'Exit Zona 1ª maquina' que liberta a zona ocupada. Por último, o *token* é descartado na atividade 'Sink'.

A atividade 'Breath' provoca um compasso de espera de 0,0 segundos. Serve para evitar que existam sobreposições de *tokens* durante o processamento de forma a garantir que eventos que ocorram simultaneamente sejam processados de forma correta. Como o tempo de espera é de 0,0 segundos, não irá afetar os tempos da linha de produção.

	A	B	C	D	E
1	Release Date	Name	Quantity	Tipo	
2	0	Product	1	1	
3	0	Product	1	2	
4	0	Product	1	3	
5					

Figura 7 - Exemplo de um plano de produção

1.2 Processamento + transporte próxima estação

O fluxograma da figura 8 cumpre as funções de controlar os tempos de processamento de cada peça em cada máquina e transportar as peças para as estações adjacentes. À medida que o modelo de simulação foi evoluindo, o controlo dos tempos de processamento foi retirado deste fluxograma. Sendo assim, vamos explicar o funcionamento do transporte das peças para as diferentes estações.

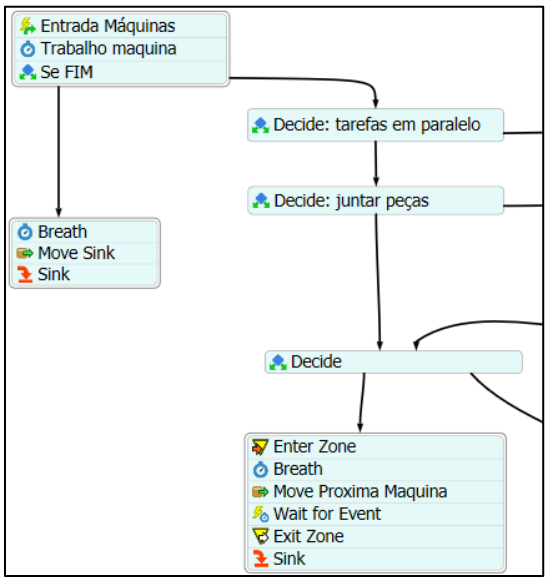

Figura 8 - *Process Flow*, 'Processamento + transporte próxima estação'

Cada vez que uma peça termina de ser processada numa máquina, o *token* associado a essa peça entra na atividade 'Se FIM' que vai verificar se essa peça deve ser transportada para outra estação de trabalho, ou se a peça já está terminada. Se for este o caso, a peça é transportada para a *Sink* do modelo 3D. Caso contrário, o *token* vai para a atividade "Decide: tarefas em paralelo".

Esta atividade verifica se a próxima estação de trabalho da peça que está associada ao *token* é uma estação com máquinas paralelas, isto é, o par de máquinas WS2_2_1 e WS2_2_2. Se isto acontecer, o *token* é movimentado para a atividade 'Enter zone' do fluxograma apresentado na secção 1.3.

Se a próxima estação de trabalho não for nenhuma das estações acima mencionadas, então o *token* avança para a atividade "Decide" que vai verificar se a próxima estação de trabalho corresponde com alguma das máquinas do grupo WS2_8 ou WS2_9. É feita esta distinção porque o controlo de estas máquinas é ligeiramente diferente do resto das máquinas.

Se a próxima estação de trabalho também não for nenhuma máquina das acima mencionadas, então o *token* é movimentado para a atividade 'Enter zone', onde tenta aceder a zona da máquina de trabalho onde vai ser feito o próximo processamento. Uma vez dentro da zona, é feito o transporte da peça até à próxima estação. Posteriormente o *token* aguarda que a peça seja retirada da estação para poder libertar a zona que foi ocupada.

Na tabela 3 podemos observar como eram obtidos os tempos de processamento de cada peça em cada estação de trabalho no modelo de simulação preliminar. A coluna 'EstaçãoParalela' indica as máquinas que trabalham em paralelo, isto é, onde o produto é dividido.

Mais à frente vão ser apresentadas as mudanças que foram feitas a este fluxograma, assim como o resultado final e o seu funcionamento.

Tabela 3 - Tabela 'tarefas'

Tipo	Estacao	ProcessingTime	EstacaoParalela
1	WS1_1	7	0
1	WS1_2	16.92	0
1	WS1_3	1	0
1	WS1_4	1	0
1	FIM	0	0
2	WS1_5	305.58	0
2	WS1_6	686.70	0
2	WS1_7	225.90	0
2	WS1_8	1	0
2	WS1_9	1	0
2	FIM	0	0
3	WS2_1	4.50	0
3	WS2_2_1	98.22	2
3	WS2_2_2	303.27	1
3	WS2_3	285.90	0
3	WS2_4	72	0
3	WS2_5	357.15	0
3	WS2_6	66	0
3	WS2_7	357.15	0
3	WS2_8_1	390	0
3	WS2_9_1	256.77	0
3	FIM	0	0

1.3 Processamento máquinas paralelas

Quando o processamento de uma peça ocorre em máquinas paralelas, o *token* associado a essa peça segue para o fluxograma 'Processamento máquinas paralelas' (figura 9). Nesta situação, uma peça é dividida em duas peças distintas, que são processadas em máquinas distintas e que no final se juntam dando lugar a uma única peça mais complexa. Vamos ver como é que este processo é tratado pelo nosso modelo.

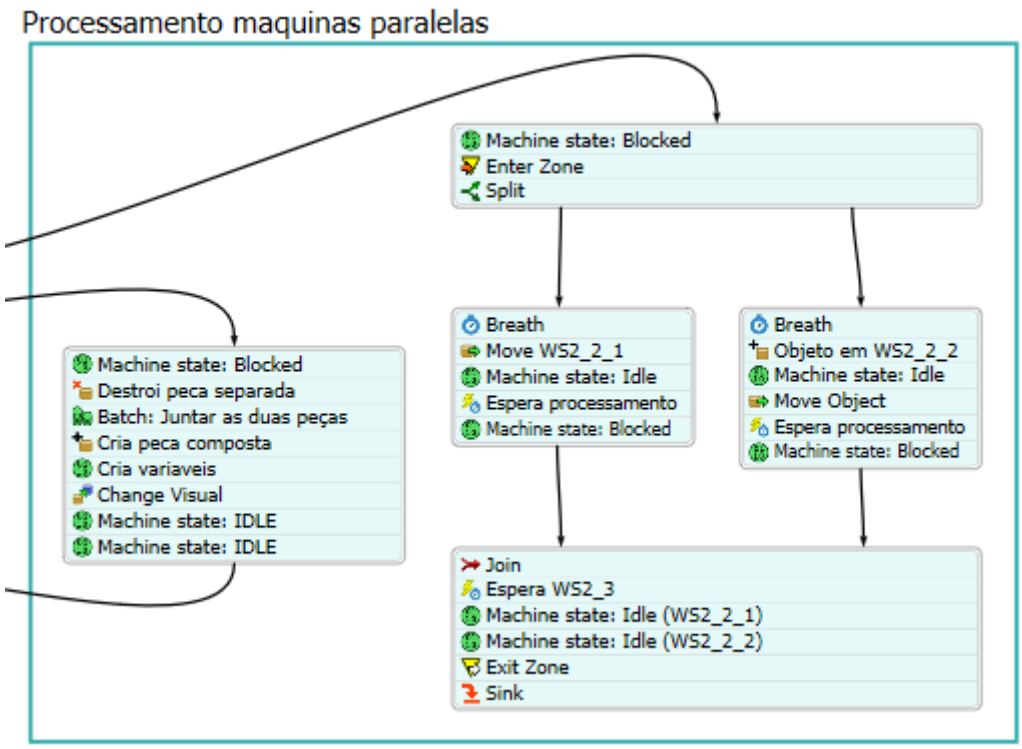

Figura 9 – *Process Flow*, 'Processamento máquinas paralelas'

Em primeiro lugar, o *token* correspondente entra numa zona que controla a capacidade do conjunto das máquinas paralelas. De seguida esse *token* é dividido em 2, na atividade 'Split'. Como resultado dessa divisão aparecem dois *tokens*, um que é o *token* original, associado à peça original, e outro que vai dar lugar a uma nova peça no modelo 3D.

Cada um dos *tokens* resultantes segue então por um ramo do fluxograma. No ramo esquerdo é feito o transporte da peça da máquina anterior para a máquina WS2_2_1.

No ramo direito é criada uma nova peça na máquina WS2_2_2, que representa a divisão do produto inicial em dois subprodutos.

Em ambos os ramos do fluxograma, cada *token* espera pelo processamento da peça, na atividade 'Espera Processamento'. O primeiro *token* a ser processado avança até a atividade 'Join', onde aguarda a chegada do *token* paralelo. Uma vez que os dois *tokens* se encontram na mesma atividade, são unidos derivando um único *token* que vai esperar a que o processamento de uma peça na máquina seguinte comece. Quando uma nova peça começar a ser processada na máquina seguinte, significa que podemos libertar a zona ocupada no início do fluxograma. Sendo assim, o *token* avança para a atividade 'Exit Zone' e 'Sink'

Como foi explicado na secção 1.2, sempre que uma peça chega a uma máquina é criado um *token* relativo a essa peça nessa máquina. Quando consideramos a questão do trabalho em máquinas paralelas, já foi apresentado como a atividade 'Split' vai gerar dois *tokens* que por sua vez vão alocar uma peça em cada máquina do processamento em paralelo. A chegada destas peças a cada máquina vai criar dois *tokens* no fluxograma 'Processamento + transporte próxima máquina'.

É o controlo desses *tokens* que permite juntar as duas peças que são trabalhadas em paralelo. Ao finalizar o processamento dessas peças a atividade 'Decide: juntar peças' presente

no fluxograma anterior deteta quando é que um *token* esta associado a uma peça que faz parte de um conjunto, e dirige esse *token* para a atividade 'Batch: Juntar as duas peças', que aguarda a chegada de outro *token* que faça parte de uma peça composta. Neste momento, os dois *tokens* criados no processamento em paralelo já foram destruídos, assim como as peças associadas. Então a atividade 'Cria peça composta' cria uma nova peça 3D na máquina seguinte e atribui-lhe as variáveis que pertenciam às peças que foram destruídas anteriormente.

Por fim, todas as atividades definidas como 'Machine State' servem para alterar as variáveis que definem o estado das máquinas. Estas atividades são necessárias para garantir que os dados apresentados nos *dashboards* correspondem com a realidade do processo.

2. Modelo final

No modelo final do processo, o trabalho a realizar em cada estação de trabalho está composto por várias tarefas, pelo que foi necessário ajustar o modelo de forma a integrar esta funcionalidade. Para isso foi atualizado o *Process Flow* adicionando um novo fluxograma que vai controlar todo o processo de trabalho das peças em cada máquina, retirando essa função do fluxograma 'Processamento + Transporte próxima estação'. Vamos então apresentar o modelo final de simulação.

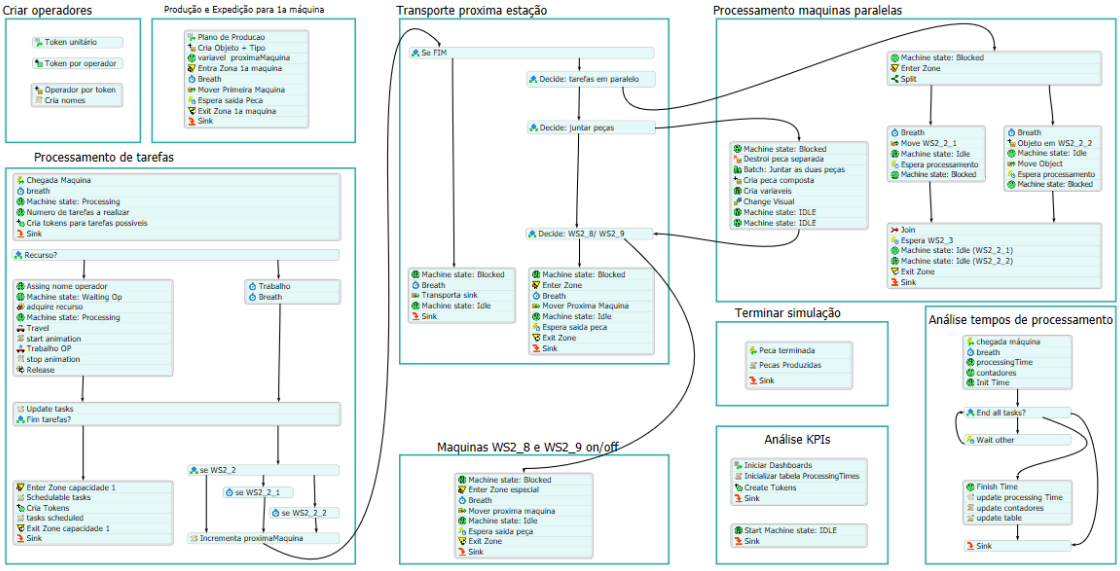

Figura 10 – *Process Flow* final

2.1 Processamento de tarefas

Um novo fluxograma foi criado de forma a poder lidar com as diversas tarefas que devem ser realizadas a cada peça em cada estação (figura 11). A ideia principal é criar tantos *tokens* como tarefas a serem realizadas, onde cada *token* contém a informação relativa a essa tarefa. Uma vez que todos os *tokens* de uma peça foram processados, essa peça está pronta para seguir para a próxima estação de trabalho.

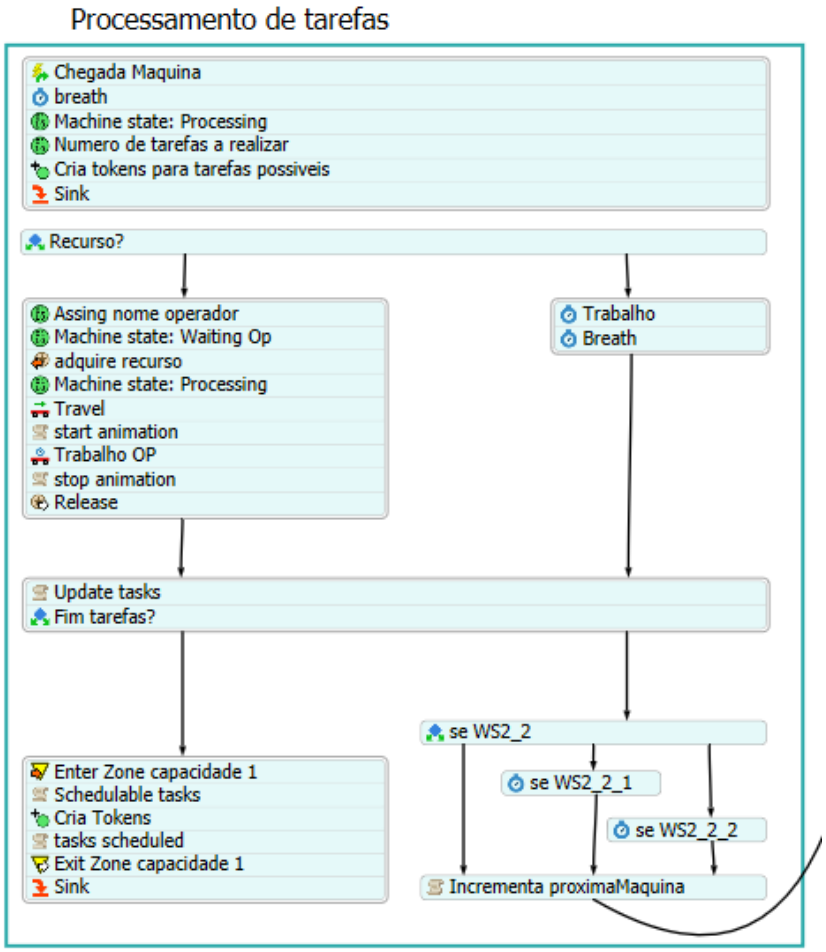

Figura 11 - *Process Flow*, 'Processamento de tarefas'

Como foi mencionado anteriormente, cada vez que uma peça chega a uma máquina, é criado um *token* que guarda qual a máquina de chegada. Isto acontece na atividade 'Chegada maquina'.

Posteriormente, na atividade 'Numero de tarefas a realizar' são criadas três variáveis associadas ao *token* criado: 'numTasks', 'Tasks' e 'currentTask'.

A variável 'numTasks' contém o número de tarefas que devem ser realizadas na peça em questão na máquina em que se encontra. Este valor é obtido consultando a tabela 'Tasks' (tabela 4).

A variável 'currentTask' serve como um contador do número de tarefas que faltam por realizar na peça e máquina em questão.

A variável 'Tasks' organiza as tarefas a realizar numa determinada peça numa estrutura. Consultando a tabela 'Tasks' é criado um vetor com tantos elementos como tarefas a realizar. Cada um desses elementos é por sua vez um vetor de 7 elementos organizados da seguinte forma:

values[i] = [0, 0, "string", "string", 0, 0, 0]

Cada um dos elementos *values* representa uma tarefa a realizar e a informação armazenada no vetor é informação relativa a essa tarefa em específico.

O primeiro valor é o 'task id' dessa tarefa, seguido pelo 'processing time' (tempo de processamento) e o operador humano que executa essa tarefa. Se tal não existir é apresentado o valor '0;'. A seguinte "string" é uma lista das tarefas precedentes, que necessitam ser realizadas antes da tarefa em questão. Por último são guardados três número inteiros (1 = sim; 2 = não) que representam se essa tarefa pode ser 'calendarizada' ou não, se já foi feita ou não e se já foi 'calendarizada' ou não. Nesta fase, o sexto e sétimo valores do vetor de cada tarefa apresentam o valor 0, pois nenhuma tarefa foi realizada. No entanto, as tarefas que possam ser realizadas logo no início deverão ter o valor 1 na quinta posição do vetor.

Uma vez que todas as variáveis foram definidas, na atividade 'Cria *tokens* para tarefas possiveis' são criados tantos *tokens* como tarefas podem ser realizadas na primeira fase do processamento da peça. Para uma tarefa poder ser realizada deve apresentar o valor 1 no quinto valor do vetor da tarefa ('schedulable') e o valor 0 no sétimo valor do vetor ('scheduled'). Significa que essa tarefa ainda não recebeu a ordem para ser executada, mas pode ser executada.

Cada um dos *tokens* criados anteriormente são colocados na atividade 'Recurso?', que analisa se cada uma das tarefas necessita um operador humano ou não (terceiro valor no vetor de cada tarefa). Se não for necessário um operador, é realizado um 'Delay' que representa o processamento da peça. Se for necessário um recurso, este é requisitado, feito o processamento da peça e posteriormente liberado. Cada recurso apenas é requisitado por uma tarefa de cada vez, pelo que se várias tarefas necessitarem do mesmo recurso, devem esperar a que este fique livre.

Terminado o processamento das tarefas são atualizados os valores das variáveis 'currentTask', que é decrementado em uma unidade, e o sexto valor do vetor da tarefa, correspondendo com a conclusão de dita tarefa. Posteriormente, na atividade 'Decide' é verificado se existe mais alguma tarefa a realizar nessa estação de trabalho. Se existirem mais tarefas a serem realizadas, é seguida a mesma lógica que foi apresentada no início. Primeiro atualizamos quais as tarefas que podem ser executadas ('schedulable'), de seguida criamos tantos *tokens* como tarefas podem ser realizadas e enviamos esses *tokens* para a atividade 'Recurso?', seguindo o percurso já apresentado. O único detalhe diferente nesta fase do processamento é que este último grupo de atividades aparece limitado por uma zona 'ZoneSchedule', de capacidade 1. Desta forma, limitamos o número de *tokens* a 1 para evitar que sejam feitas alterações dos valores das variáveis de forma incorreta.

Tabela 4 - Excerto da tabela 'Tasks'

Workstation	Tipo	Task - ID	Resource	Processing Ti	Precedences
WS1_1	1	10	Op1	0.20	0
WS1_1	1	20	Op1	0.80	10
WS1_1	1	30	Op1	0.90	20
WS1_1	1	40	Sem Operadc	1	0
WS1_1	1	50	Sem Operadc	6	40
WS1_1	1	55	Sem Operadc	0.55	40
WS1_1	1	60	Sem Operadc	2	40
WS1_1	1	65	Sem Operadc	0.55	40
WS1_1	1	70	Op1	0.22	40
WS1_1	1	80	Op1	0.25	70
WS1_1	1	90	Op1	2	80
WS1_1	1	100	Op1	0.50	90

Se não existirem mais tarefas a serem realizadas, a atividade 'Fim tarefas?' encaminha o último *token* que foi processado para a atividade 'Incrementa proximaMáquina' que vai atualizar a variável responsável de indicar qual a próxima estação de trabalho para a peça que terminou de ser processada.

As atividades de decisão e *delay* implementadas nesta fase servem para libertar as máquinas de trabalho paralelo, caso estas tenham sido utilizadas.

2.2 Máquinas WS2_8 e WS2_9 on/off

O bloco de atividades da figura 12 foi adicionado de forma a poder controlar o estado de funcionamento dos pares de máquinas WS2_8 (1 e 2) e WS2_9 (1 e 2). Se ambas as máquinas do par estiverem ligadas, então podem ser enviadas peças para uma ou outra máquina em função da capacidade disponível em cada uma. Se alguma das máquinas estiver desligada só a outra poderá receber peças para serem processadas. O estado das máquinas é definido na tabela 'Maquinas' (tabela 5).

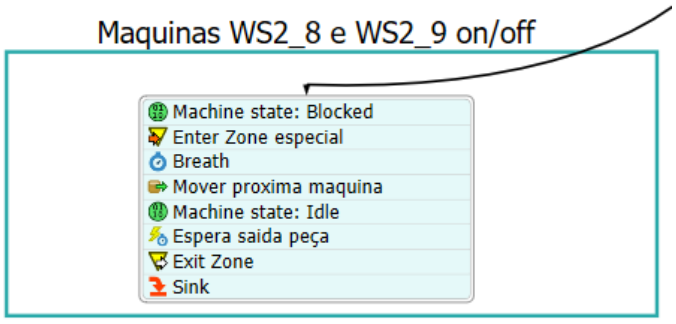

Figura 12 – *Process Flow*, 'Máquinas WS2_8 e WS2_9 on/off'

Tabela 5 - Tabela 'Máquinas'

Maquina	Ligado
WS1_1	1
WS1_2	1
WS1_3	1
WS1_4	1
WS1_5	1
WS1_6	1
WS1_7	1
WS1_8	1
WS1_9	1
WS2_1	1
WS2_2_1	1
WS2_2_2	1
WS2_3	1
WS2_4	1
WS2_5	1
WS2_6	1
WS2_7	1
WS2_8_1	1
WS2_8_2	1
WS2_9_1	1
WS2_9_2	1

Tabela 6 - Nova tabela 'tarefas'

Tipo	Estacao	EstacaoParalela
1	WS1_1	0
1	WS1_2	0
1	WS1_3	0
1	WS1_4	0
1	FIM	0
2	WS1_5	0
2	WS1_6	0
2	WS1_7	0
2	WS1_8	0
2	WS1_9	0
2	FIM	0
3	WS2_1	0
3	WS2_2_1	2
3	WS2_2_2	1
3	WS2_3	0
3	WS2_4	0
3	WS2_5	0
3	WS2_6	0
3	WS2_7	0
3	WS2_8_1	0
3	WS2_9_1	0
3	FIM	0

2.3 Análise KPIs

O grupo de atividades da figura 13 inicializa as diferentes variáveis necessárias para fazer a correta recolha dos KPIs da planta de produção.

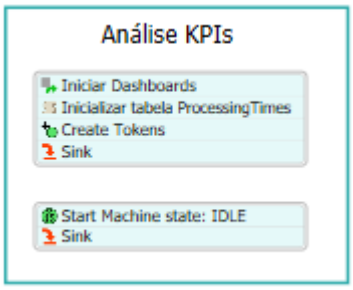

Figura 13 – *Process Flow*, 'Análise KPIs'

Na atividade 'Inicializar tabela ProcessingTimes' todos os valores da tabela referida são colocados a zero. Essa tabela vai apresentar os tempos de processamento médios, mínimos e máximos em cada uma das estações de trabalho.

Posteriormente, na atividade 'Create Tokens' são criados tantos *tokens* como estações de trabalho existem na planta de produção. Cada um desses *tokens* vai inicializar o estado de cada uma das máquinas na atividade 'Start Machine state: IDLE'. Desta forma garantimos que a utilização das máquinas começa sempre como livre.

2.4 Análise tempos de processamento

O fluxograma da figura 14 cumpre a função de controlar os tempos de processamento de cada uma das estações de trabalho. Como foi referido anteriormente, esses tempos de funcionamento serão apresentados ao utilizador utilizando a tabela 'ProcessingTimes'.

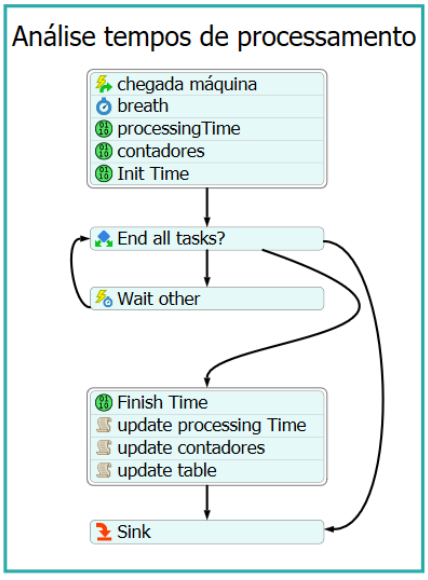

Figura 14 – *Process Flow*, 'Análise tempos de processamento'

Cada vez que uma peça chega a uma máquina é criado um *token* na atividade 'chegada maquina'. A esse *token*, que guarda o nome da máquina onde acabou de entrar, são adicionadas

três variáveis: o tempo de processamento associado, o número de peças que já passaram por essa máquina e o tempo de processamento acumulado nessa máquina. Sempre que uma peça entra numa máquina começa de imediato o processamento, logo na atividade 'Init Time' é guardado o tempo de início do processamento da peça.

De seguida é verificado se a peça que se encontra na estação de trabalho associada ao *token* já terminou de ser processada. Se ainda faltarem tarefas a realizar, o *token* avança para a atividade 'Wait other' onde aguarda que o estado da máquina seja alterado. Cada vez que isto acontece voltamos a verificar se existem mais tarefas a serem realizadas na peça em questão.

Se já não existir mais nenhuma tarefa a ser realizada, então é guardado o tempo de fim de processamento na atividade 'Finish Time'. Posteriormente são calculados os tempos de processamento e atualizados os valores que vão ser apresentados na tabela 'ProcessingTimes'.

2.5 Criar operadores

Um problema que o nosso modelo de simulação apresentava numa primeira fase estava relacionado com a criação dos operadores. Em função do modelo de otimização utilizado, a simulação podia necessitar mais ou menos operadores, que deviam ser criados ou destruídos.

Figura 15 – *Process Flow*, 'Criar operadores'

Para solucionar este problema foi criado o fluxograma da figura 15 que permite criar os operadores da planta de produção de forma dinâmica, em função dos dados de entrada.

A atividade 'Token unitário' apenas cria um *token* no início da simulação. Quando não está ligada a mais nenhuma atividade, nada é alterado no nosso modelo. Mas se for unida a atividade 'Token por operador', então são criados tantos *tokens* como operadores foram definidos no modelo de otimização. Cada um desses *tokens* são criados na atividade 'Operadores por token' que vão criar um elemento 3D no modelo. Neste caso um operador. Finalmente são atribuídos nomes a cada um dos operadores usando uma lógica que une o nome da estação em que cada operador está alocado com o número do operador.

A união das duas primeiras atividades só deve ser feita uma vez, de forma a que sejam criados todos os operadores necessários para correr o modelo de simulação. Uma vez criados todos os operadores, estes devem ser adicionados ao grupo 'CreatedOperators'.

Para eliminar todos os operadores da fábrica, por exemplo para correr outra simulação com dados de entrada diferentes, foi criado um código inserido num elemento 'Model Trigger'.

Este código atua sempre que é feito *Reset* no modelo e elimina todos os operadores que existem no modelo no momento do *Reset*. O código encontra-se comentado, e apenas deve ser descomentado quando seja necessário eliminar todos os operadores da planta de produção.

2.6 Terminar simulação

O conjunto de atividades da figura 16 apenas permite terminar a execução do modelo de simulação. Implementa um contador que é incrementado cada vez que uma peça é depositada na *Sink* do modelo. Quando o contador atinge o total das peças enviadas para o plano de produção, o modelo de simulação é terminado. A contagem das peças é armazenada numa variável global.

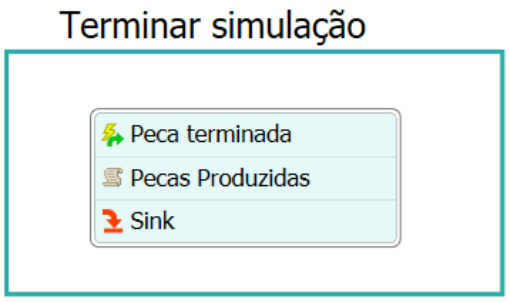

Figura 16 – *Process Flow*, 'Terminar simulação'

3. *Dashboards*

Foram criadas várias páginas de *dashboards* onde é possível observar os KPIs do processo produtivo. Estas páginas recolhem a informação dos *tokens* que percorrem o *Process Flow*.

3.1 Peças produzidas

Faz a contagem do número de peças produzido por cada estação de trabalho, e apresenta a informação na tabela 7:

Tabela 7 - KPI, Peças produzidas

Object	Throughput
WS1_1	2577.00
WS1_2	2577.00
WS1_3	2577.00
WS1_4	2577.00
WS1_5	63.00
WS1_6	63.00
WS1_7	63.00
WS1_8	63.00
WS1_9	63.00
WS2_1	110.00
WS2_2_1	110.00
WS2_2_2	110.00
WS2_3	110.00
WS2_4	110.00
WS2_5	110.00
WS2_6	110.00
WS2_7	110.00
WS2_8_1	55.00
WS2_8_2	55.00
WS2_9_1	110.00

3.2 Tempo em cada máquina

Conta o tempo que cada peça passa em cada estação, desde que entra para ser processada até que é transportada para a próxima estação. Esta informação é apresentada na tabela 8, onde é possível observar o tempo médio de permanência em cada máquina, assim como o tempo mínimo e máximo.

Tabela 8 - KPI, Tempos de permanência

Object	Avg Staytime	Min Staytime	Max Staytime
WS1_1	1014.99	420.00	1030.20
WS1_2	1015.22	1015.20	1030.20
WS1_3	0.00	0.00	0.00
WS1_4	0.00	0.00	0.00
WS1_5	40645.72	11730.60	41117.93
WS1_6	41112.09	41112.00	41117.93
WS1_7	4320.13	4320.00	4327.95
WS1_8	0.00	0.00	0.00
WS1_9	0.00	0.00	0.00
WS2_1	15315.63	203.56	15930.00
WS2_2_1	4275.06	4275.00	4281.66
WS2_2_2	6240.67	6240.60	6248.10
WS2_3	15546.61	8640.00	15930.00
WS2_4	15455.72	4320.00	15930.00
WS2_5	15632.00	4590.00	15930.00
WS2_6	15778.90	3960.00	15930.00
WS2_7	15930.15	15930.00	15946.32
WS2_8_1	23130.33	23130.00	23147.95
WS2_8_2	23130.34	23130.00	23148.56
WS2_9_1	3240.18	3240.00	3260.32
WS2_9_2	0.00	0.00	0.00

3.3 Tempo de produção

Conta o tempo que cada peça gasta em cada estação a ser processada, seja por um operador humano ou por um robô. Este tempo nunca é superior ao tempo de permanência em cada estação. Esta informação é apresentada na tabela 9, onde é possível observar o tempo médio de processamento, assim como os tempos máximos e mínimos.

Tabela 9 - KPI, Tempos de produção

Maquina	AvgProcessingTime	MinProcessingTime	MaxProcessingTime
WS1_1	420	420	420.00
WS1_2	1015.20	1015.20	1015.20
WS1_3	0	0	0
WS1_4	0	0	0
WS1_5	18514.80	18514.80	18514.80
WS1_6	41202	41202.00	41202
WS1_7	13554	13554	13554
WS1_8	0	0	0
WS1_9	0	0	0
WS2_1	270	270	270
WS2_2_1	5908.80	5898.60	5938.20
WS2_2_2	42658.20	42658.20	42658.20
WS2_3	34308.00	34308	34308.00
WS2_4	4320	4320	4320
WS2_5	17541.00	17541	17541.00
WS2_6	3960	3960	3960
WS2_7	15930	15930	15930
WS2_8_1	27091.80	27091.80	27091.80
WS2_8_2	0	0	0
WS2_9_1	15454.80	15454.80	15454.80
WS2_9_2	0	0	0

3.4 Utilização das máquinas

É contado o tempo que cada estação de trabalho passa em cada um dos três estados possíveis: livre, a processar ou bloqueada. Quando uma máquina se encontra "livre" significa que não tem nenhuma peça para processar. Quando recebe uma peça, passa ao estado "a processar" até que termina todas as tarefas indicadas para essa peça nessa estação. Se a máquina não conseguir enviar a peça porque a estação seguinte se encontra ocupada, então esta estação passa para o estado "bloqueada". A informação é apresentada num gráfico circular como o da figura 17, onde a cor verde significa "livre", azul significa "a processar" e vermelho significa "bloqueado".

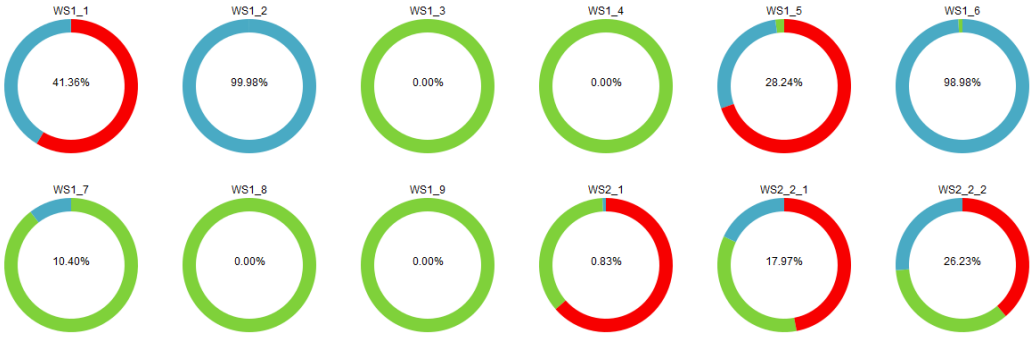

Figura 17 - KPI, Utilização das máquinas

3.5 Utilização dos operadores

Conta o tempo que cada operador passa num de três estados possíveis: livre, a processar e em viagem. O estado "livre" acontece quando um operador não está a efetuar nenhum trabalho. O tempo que um operador precisa de se deslocar para uma estação quando vai começar a processar uma peça é considerado tempo "em viagem". E quando um operador se encontra a trabalhar, está no estado "a processar". A informação é apresentada num gráfico circular como o da figura 18, onde a cor verde significa "livre", azul significa "a processar" e amarelo significa "em viagem".

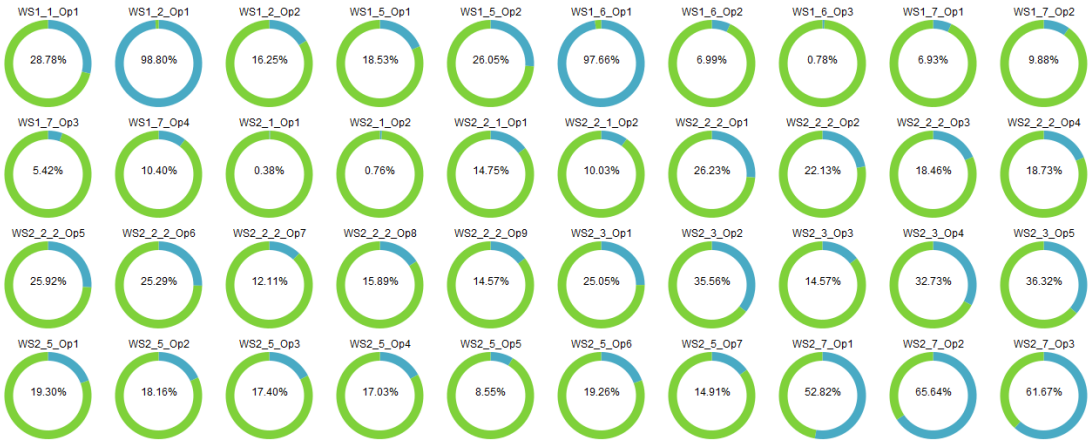

Figura 18 - KPI, Utilização dos operadores

Uma vez que foram apresentados todos os elementos constituintes do modelo de simulação, passamos então à descrição do modelo de otimização.

Capítulo 5 – Otimização

Aliado ao modelo de simulação apresentado foi desenvolvido um modelo matemático capaz de calcular a melhor opção de alocação de recursos para cada estação de trabalho. O modelo matemático atribui a cada operador alocado uma série de tarefas a realizar numa determinada ordem de forma a que o tempo de produção (*makespan*) seja mínimo.

Este modelo matemático foi introduzido num programa de otimização que foi desenvolvido de forma a permitir otimizar mais de uma estação ao mesmo tempo. O modelo matemático desenvolvido é implementado em dois *softwares* diferentes, CPLEX e OR Tools. Um dos objetivos da dissertação passa por fazer uma comparação destes dois *softwares* pelo que a linguagem escolhida para desenvolver o programa de otimização foi a linguagem *Python*. Desta forma a implementação dos dois *softwares* é muito semelhante, o que permite uma comparação mais simples.

1. Modelo matemático de otimização

O modelo matemático utilizado foi definido em [10] e permite obter a alocação de tarefas aos operadores de forma a que o tempo de conclusão de todas as tarefas seja o menor possível. De seguida vão ser apresentados os parâmetros, variáveis de decisão, restrições e a função objetivo do modelo utilizado.

1.1 Parâmetros

O modelo implementado apresenta os seguintes parâmetros:

T: conjunto de tarefas a realizar;

TP_i: conjunto de tarefas precedentes de i;

TNO: conjunto de tarefas a ser realizadas por um operador humano;

O: conjunto de operadores humanos;

t_i: duração da tarefa i, $i \in T$;

$T_m = \sum t_i$, $i \in T$.

1.2 Variáveis de decisão

As variáveis de decisão consideradas pelo modelo são as seguintes:

s_i: tempo de início da tarefa i, $i \in T$;

v_{ij}: variável binária - valor 1 se a tarefa i é realizada e só depois ocorre a tarefa j; valor 0 caso contrário, $\forall_{i,j} \in TNO$;

x_{oi}: variável binária - valor 1 se o operador o realiza a tarefa i; valor 0 caso contrário, $\forall_o \in O$, $\forall_{i,} \in TNO$;

y_{oij}: variável binária - valor 1 se o operador *o* realiza as tarefas *i* e *j*; valor 0 caso contrário, $\forall_o \in O,\ \forall_{i,j} \in TNO$;

MS: makespan (variável auxiliar, representa o valor da função objetivo).

1.3 Restrições

As restrições implementadas pelo problema são as seguintes:

- Restrição 1: As relações de precedência entre duas tarefas *i* e *j* são respeitadas. Uma dada tarefa *i* só pode ser iniciada num tempo igual ou superior ao do início da tarefa *j* mais a sua duração:

$$s_i \geq s_j + t_j, \qquad \forall_i \in T,\ \forall_j \in TP_i \tag{16}$$

- Restrição 2: O *makespan* é suficientemente grande para que a tarefa com maior duração possa terminar:

$$MS \geq s_i + t_i, \quad \forall_i \in T \tag{17}$$

- Restrição 3: Duas tarefas *i* e *j* só são realizadas uma vez e numa determinada ordem: *i* é realizada e depois ocorre *j* ou ocorre *j* e depois *i*:

$$v_{ij} + v_{ji} = 1, \qquad \forall_{i,j} \in TNO \tag{18}$$

- Restrição 4: As relações de precedência entre as tarefas realizadas por operadores humanos são respeitadas:

$$s_j \geq s_i + t_i - M(1 - v_{ij}) - M\left(1 - \sum_{o \in O} y_{oij}\right), \quad \forall_{i,j} \in TNO: i \neq j \tag{19}$$

- Restrição 5: A cada tarefa que necessita de um operador humano é atribuído um único operador:

$$\sum_{o \in O} x_{oi} = 1, \quad \forall_i \in TNO \tag{20}$$

- Restrição 6: Se o operador o realiza as tarefas *i* e *j*, então a variável y_{oij} assume o valor 1, ou o valor 0 caso contrário:

$$y_{oij} \geq x_{oi} + x_{oj} - 1, \quad \forall_{i,j} \in TNO, \quad \forall_o \in O \tag{21}$$

$$y_{oij} \leq \frac{x_{oi} + x_{oj}}{2}, \quad \forall_{i,j} \in TNO, \quad \forall_o \in O \tag{22}$$

1.4 Função objetivo

Uma vez que o objetivo é minimizar o tempo de conclusão de todas as tarefas de cada estação, e que este tempo está representado pela variável MS, o objetivo é então minimizar o valor desta variável MS, ou *makespan*:

$$\min MS \tag{23}$$

1.5 Parâmetro M (bigM)

Quando foi desenvolvido o modelo de otimização foi possível observar que um dos parâmetros apresentados era o parâmetro M, que apresenta uma função muito específica, na restrição 4 do nosso problema.

$$s_j \geq s_i + t_i - M(1 - v_{ij}) - M\left(1 - \sum_{o \in O} y_{oij}\right), \quad \forall_{i,j} \in TNO: i \neq j \tag{24.1}$$

Na expressão 24.1, que representa a 4ª restrição do nosso problema podemos observar que dois parâmetros da equação dependem do valor M, assim como das variáveis binárias v_{ij} e y_{oij}.

Analisando por exemplo o primeiro desses parâmetros, que depende de v_{ij}, podemos deduzir que sempre que esta variável tiver o valor 1, a restrição vai estar ativa e esta parte da equação não vai afetar o valor final da restrição. No entanto, quando o valor da variável for 0, desativamos a restrição (o valor M define o resultado da restrição).

Na maioria dos modelos matemáticos que tratam problemas de otimização, este parâmetro M apresenta um valor muito elevado suficiente para desativar as restrições (1000 por exemplo), mas ao implementar este problema no software CPLEX observamos que obtemos erros nas soluções obtidas devido a erros de aproximação que o software assume.

Mais concretamente, quando a variável v_{ij} assume o valor 0, o CPLEX apresenta esse valor, no entanto, quando essa variável assume o valor 1 o CPLEX apresenta um valor arredondado muito próximo, por exemplo 0.9998. Sendo assim, se atribuirmos ao parâmetro M um valor muito grande, pode ocorrer que a restrição não seja ativada.

Para solucionar este problema foi necessário determinar qual o mínimo valor de M que garante que essa parcela da restrição é desativada quando $v_{ij} = 0$. Vejamos isso na primeira parte da equação 24:

$$s_j \geq s_i + t_i - M(1 - 0) \tag{24.2}$$

Logo:

$$M \geq s_i + t_i - s_j \tag{24.3}$$

Analisando o pior dos casos, s_j = 0 e $s_i + t_i$ apresenta o maior valor possível. Este valor é o da última tarefa a ser executada, pelo que podemos obter o valor de M somando os tempos de processamento de todas as tarefas a realizar. Sendo assim, o parâmetro M aparece definido nos parâmetros do modelo como o somatório de todos os tempos de processamento de todas as tarefas consideradas. Desta forma, já não ocorrem erros nas soluções obtidas.

2. Implementação do modelo

Como foi explicado anteriormente, vão ser utilizadas dois *softwares* de otimização no nosso programa, o *software* CPLEX e o *software* OR Tools, ambos implementados na linguagem de programação *Python*. O programa que foi desenvolvido lê de uma folha Excel todos os dados necessários para efetuar a otimização das estações definidas, e escreve os resultados noutra folha Excel que posteriormente pode ser utilizada pelo modelo de simulação. De seguida vão ser apresentadas a estrutura e o funcionamento do programa.

2.1 Dados de entrada e modos de funcionamento

Os dados de entrada do programa devem ser fornecidos num documento Excel com 5 folhas organizadas da seguinte forma:

1. ModoOperacao: Nesta folha é escolhido qual o modo de operação que o programa vai utilizar, assim como o tempo máximo de otimização por iteração do modelo.

2. Station_Operators: Esta folha apresenta uma lista de cada uma das máquinas a otimizar. Dependendo do modo de operação escolhido, será também necessário definir o número de operadores disponíveis em cada estação. Por último podem ser definidos valores de *upper bound* e *lower bound* (limite máximo e mínimo do makespan) para cada estação de trabalho de forma a melhorar o tempo de otimização de cada problema.

3. PrecedencesSet: Esta folha define a lista de precedências entre as tarefas de cada estação de trabalho.

4. NeedsOperators: Esta folha define a lista de tarefas de cada estação de trabalho que necessitam de um operador humano.

5. Tipo: Indica que tipo de peças são produzidas em cada estação de trabalho

Todos os dados recolhidos são armazenados numa estrutura interna do programa que aparece representada na figura 21 como 'SistemData' e de seguida é iniciada a ferramenta de otimização num de dois modos de funcionamento possíveis:

Modo 1: Neste modo o programa irá utilizar todos os operadores disponíveis, previamente definidos no documento Excel, na folha 'Station_Operators', de forma a encontrar a melhor solução possível. Se tiverem sido definidos *upper bound* e/ou *lower bound*, estes também vão ser incluídos na ferramenta de otimização de modo a acelerar o modelo.

Modo 2: Neste modo o programa irá percorrer cada estação de trabalho tentando procurar o número ótimo de operadores que devem ser alocados por estação. No fluxograma da figura 19 podemos observar o funcionamento deste modo de operação.

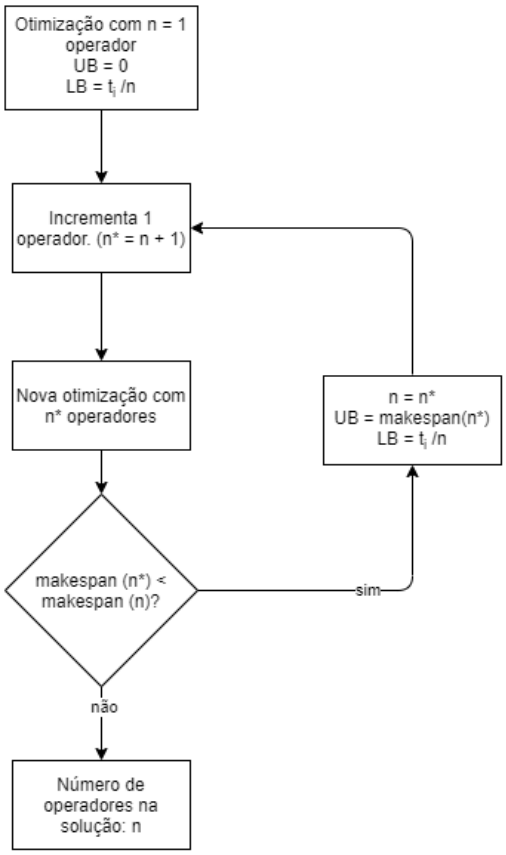

Figura 19 – Diagrama de funcionamento do modo 2 de otimização

Em primeiro lugar é efetuada a otimização da estação com um operador. De seguida é incrementado o número de operadores alocados à estação de trabalho e é feita uma nova otimização. São então comparadas as soluções da otimização. Se a solução com mais operadores apresentar um *makespan* inferior são atualizados os valores de *upper bound* e *lower bound* e de seguida é feita uma nova otimização com mais um operador alocado. Se a solução não apresentar um *makespan* inferior temos o número ótimo de operadores que vão proporcionar o tempo de produção mínimo para essa estação de trabalho. Em cada iteração os valores de *upper bound* e *lower bound* são definidos seguindo as seguintes regras:

- *Upper bound*: Valor da solução anterior, isto é, com um operador menos. Durante a primeira iteração o problema não inclui nenhum *upper bound*.

- *Lower bound*: Somatório dos tempos de processamento das tarefas da estação a ser otimizada sobre o número de operadores incluídos.

$$LB = \frac{\sum t_i}{número\ de\ operadores}, \quad i \in T \tag{25}$$

Assim que todas as estações tenham sido otimizadas, é invocada a função *calculate_tasks_excecuters()* que irá definir, no 'SistemData', a alocação dos operadores a cada uma das tarefas a realizar seguindo a otimização feita pela ferramenta. Além disso, esta função irá criar uma série de precedências fictícias, quando for necessário, para garantir que não existe nenhum tipo de ambiguidade na ordem das operações quando estas forem enviadas para o modelo de simulação.

2.2 Precedências fictícias

As precedências fictícias ocorrem quando o *software* de otimização determina que um operador deve realizar duas tarefas consecutivas, que não têm nenhum tipo de precedência entre elas. Nesta situação, o programa verifica na solução encontrada pelo *software* de otimização qual é a tarefa que é realizada antes e introduz uma precedência entre essas duas tarefas. Desta forma é garantido que a ordem definida pelo *software* de otimização é cumprida na simulação.

Consideremos o exemplo onde um operador necessita realizar 4 tarefas distintas (A, B, C e D). Essas tarefas apresentam umas regras de precedência como as que estão indicadas na figura 20:

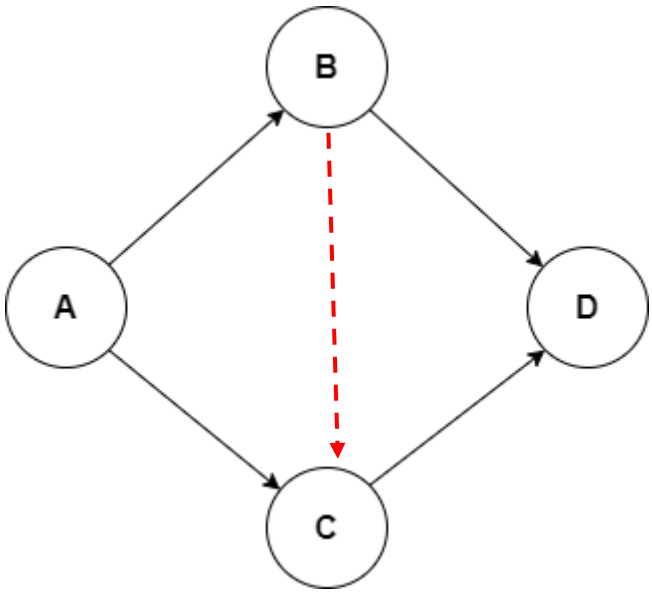

Figura 20 - Diagrama de precedências, com uma precedência fictícia

Analisando esta situação podemos perceber que a primeira tarefa a ser executada será a tarefa 'A', pois é a única que não tem precedências anteriores. De seguida, existem duas

tarefas que podem ser realizadas por qualquer ordem, a tarefa 'B' e 'C'. Sendo assim, devemos criar uma precedência fictícia que imponha uma prioridade neste tipo de situações, representada a vermelho na figura 20. Com isto conseguimos garantir que a tarefa 'B' será realizada sempre antes da tarefa 'C', ficando clara qual a sequência de tarefas que cada operador deve efetuar.

2.3 Diagrama UML

O diagrama UML da figura 21 representa a estrutura do programa de otimização desenvolvido.

Podemos observar como a estrutura do programa desenvolvido é comum para os dois *softwares* utilizados, o que permite que as mesmas estruturas de dados sejam utilizadas por ambas as ferramentas.

Quando o utilizador introduz os dados de otimização, utilizando um documento Excel, a classe 'ExcelRead' lê os dados introduzidos e organiza-os seguindo a estrutura definida na classe 'SistemData'. Esta classe contém um dicionário com todas as tarefas que foram lidas no documento Excel. Por sua vez, essas tarefas são guardadas noutra classe de forma a poderem englobar toda a informação que lhes é relacionada: tempos de processamento, tarefas com necessidade de operadores e relações de precedência.

De seguida, quando o programa chama a função '*optimize_data()*' é criado o modelo de otimização para a estação definida, utilizando um dos *softwares* de otimização disponíveis. O modelo de otimização vai então encontrar uma solução de alocação para o problema definido, e definir essa solução no 'SistemData' do programa.

Por fim a classe 'ExcelWrite' escreve num documento Excel a lista de tarefas e alocações resultantes do processo de otimização para cada estação que foi otimizada. A estrutura deste documento Excel encontra-se definida no capítulo 4, e permite que os dados sejam utilizados no modelo de simulação já apresentado.

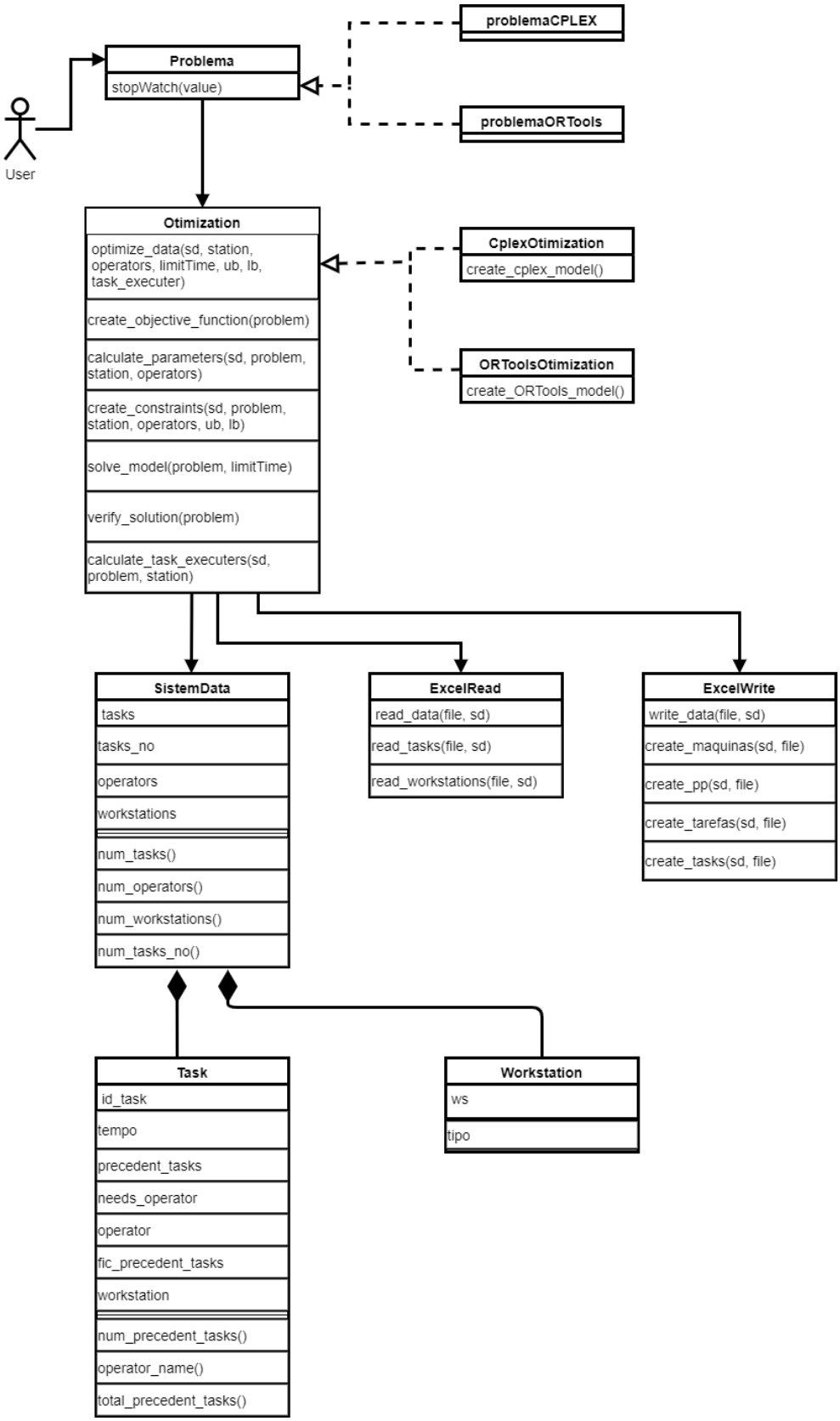

Figura 21 – Diagrama UML do modelo de otimização

3. Ferramentas de otimização

Este trabalho pretende fazer uma comparação entre dois softwares de otimização, CPLEX e OR Tools. Para isso, a ferramenta de otimização desenvolvida disponibiliza duas versões diferentes, uma com cada um dos *softwares* indicados.

O CPLEX, da empresa IBM, é um *software* com uma capacidade computacional muito elevada. Consegue proporcionar resultados satisfatórios em intervalos de tempo aceitáveis e ainda possui uma sintaxe de programação intuitiva e com grande suporte tanto em páginas online como no site oficial. O único problema que apresenta é que a versão gratuita tem um limite de 1000 variáveis e 1000 restrições por problema, o que num problema da dimensão do que está a ser tratado aqui é facilmente alcançável. Se quisermos obter uma versão completa desta ferramenta deveremos adquirir uma licença de utilização. Os valores destas licenças começam nas centenas de euros por mês, para profissionais cujo desenvolvimento não envolva a utilização de servidores e podem aumentar consideravelmente para outro tipo de aplicações.

Por outro lado, o OR Tools, da empresa Google, já se apresenta como um *software open source*, o que significa que é gratuito e que qualquer pessoa ou empresa o pode instalar na máquina que pretenda utilizar. No entanto este produto é relativamente recente no mercado, pelo que não dispõe de um suporte tão diversificado como o do CPLEX, e não é tão poderoso em termos computacionais, pelo que em grandes problemas pode apresentar alguma dificuldade em encontrar resultados satisfatórios.

No próximo capítulo serão apresentados os resultados obtidos e será feita uma comparação para cada um dos *softwares* de otimização apresentados.

Capítulo 6 – Resultados

Neste capítulo vão ser apresentados dois cenários de análise que nos vão permitir avaliar os resultados do trabalho desenvolvido.

Em cada uma das secções seguintes vai ser apresentado o cenário em análise, os resultados obtidos com cada *software* de otimização, a validação dos resultados obtidos e uma comparação do desempenho entre os dois tanto a nível de soluções de alocação propostas como de KPIs apresentados no modelo de simulação.

1. Cenário 1 - Tempo mínimo de produção

1.1 Descrição do cenário de análise

No primeiro cenário a analisar queremos obter o menor tempo de ciclo em cada estação. Para isso não foi imposto nenhum limite de operadores a cada estação de trabalho.

O programa de otimização foi utilizado no modo 2 de funcionamento, onde é encontrado o número ótimo de operadores a ser alocados, de forma a que a solução obtida por cada uma das ferramentas fosse a melhor solução possível de encontrar.

Ambos os problemas têm a mesma lista de tarefas, com os mesmos tempos de processamento e o tempo limite dedicado à procura de uma solução, por máquina por operador foi limitado a 10 minutos. Ao fim de 10 minutos de processamento o programa deve retornar a última solução que conseguiu encontrar, independentemente de esta ser ótima ou não.

Foi utilizado um plano de produção com 2750 ordens de produção, distribuídas como indicado na tabela 10:

Tabela 10 – Ordens de produção

Peça	*Quantidade*
Tipo 1	2577
Tipo 2	63
Tipo 3	110

1.2 Resultados da otimização

Na tabela 11 podemos observar os resultados obtidos no modelo de otimização para as especificações do cenário 1. São apresentados os operadores alocados na melhor solução encontrada por cada *software* assim como o tempo de ciclo de cada estação de trabalho e o tempo que cada ferramenta demorou a encontrar essa solução. Podemos observar algumas diferenças de alocação de operadores nas estações WS1_7, WS2_2_2, WS2_3, WS2_8 e WS2_9.

Tabela 11 - Resultados da otimização, cenário 1

	OR Tools			*CPLEX*		
Máquina	Operadores	*Makespan (min)*	Tempo de execução	Operadores	*Makespan (min)*	Tempo de execução
WS1_1	1	7	0.2301	1	7	0.359
WS1_2	2	16.92	0.7719	2	16.92	0.6911
WS1_3	0	0	0.0359	0	0	0.0528
WS1_4	0	0	0.0359	0	0	0.0498
WS1_5	2	195.51	7.7752	2	195.51	1.5358
WS1_6	3	685.2	8.9391	3	685.2	1.5488
WS1_7	1	225.9	0.5056	1	225.9	0.5555
WS1_7	2	-	323.3665	4	72	1.5348
WS1_8	0	0	0.0418	0	0	0.0728
WS1_9	0	0	0.0698	0	0	0.01047
WS2_1	2	3.3	0.2333	2	3.3	0.3191
WS2_2_1	2	71.25	2.4055	2	71.25	1.4032
WS2_2_2	1	710.97	3.4647	1	710.97	2.5352
WS2_2_2	2	-	592.1049	9	104	612.1236
WS2_3	1	571.8	0.734	1	571.8	0.6797
WS2_3	2	-	106.6152	5	144	23.2885
WS2_4	0	72	0.0389	0	72	0.0568
WS2_5	7	76.5	228.7367	7	76.5	53.6746
WS2_6	0	66	0.0468	0	66	0.0585
WS2_7	3	265.5	142.1533	3	265.5	18.5836
WS2_8_1	3	451.53	599.7185	3	488.4	602.4878
WS2_8_1	4	-	595.7647	5	385.5	10.7328
WS2_9_1	5	61.29	599.0296	5	54	37.1008
WS2_9_1	7	54	87.1989			

1.3 Validação dos resultados no modelo de simulação

De forma a poder validar o modelo de simulação implementado, devemos correr a simulação com as soluções propostas no modelo de otimização, e verificar se os tempos de processamento em cada estação correspondem com os tempos que foram obtidos de tais otimizações. Na tabela 12 estão apresentados os tempos de processamento estimados, provenientes da otimização, e os tempos de processamento da simulação.

Tabela 12 - Validação de resultados, cenário 1

	OR Tools		*CPLEX*	
Máquina	Tempo estimado (min)	Tempo de simulação (min)	Tempo estimado (min)	Tempo de simulação (min)
WS1_1	7	7	7	7
WS1_2	16.92	16.92	16.92	16.92
WS1_3	0	0	0	0
WS1_4	0	0	0	0
WS1_5	195.51	195.51	195.51	195.51
WS1_6	685.2	685.2	685.2	685.2
WS1_7	225.9	225.9	72	72
WS1_8	0	0	0	0
WS1_9	0	0	0	0
WS2_1	3.3	3.3	3.3	3.3
WS2_2_1	71.25	72.6	71.25	71.69
WS2_2_2	710.97	710.97	104	105.53
WS2_3	571.8	571.8	144	144
WS2_4	72	72	72	72
WS2_5	76.5	76.5	76.5	76.5
WS2_6	66	66	66	66
WS2_7	265.5	265.5	265.5	265.5
WS2_8_1	451.53	451.53	385.5	385.5
WS2_8_2	451.53	0	385.5	385.5
WS2_9_1	54	56.94	54	55.91
WS2_9_2	54	0	54	0

Podemos ver como quase todos os tempos de processamento obtidos na simulação coincidem com os valores esperados. Existem alguns casos onde existe uma pequena diferença entre o valor esperado e o valor obtido na simulação, mas que podemos considerar pouco significativas. Isto permite nos validar o modelo de simulação implementado.

1.4 Comparação dos resultados

Como foi possível observar nas secções 1.2 e 1.3, a utilização de diferentes *softwares* de otimização produziu soluções com diferentes números de operadores alocados nas estações WS1_7, WS2_2_2, WS2_3, WS2_8 e WS2_9. Por isso, os tempos de processamento nestas estações também apresenta diferenças consideráveis. Nas estações WS1_7, WS2_2_2 e WS2_3 a ferramenta CPLEX conseguiu alocar mais operadores a cada estação, pelo que os tempos de processamento foram menores. E se compararmos as soluções com o mesmo número de operadores, o CPLEX continua a obter valores de *makespan* inferiores. Para a estação WS2_8, com 3 operadores alocados, a ferramenta OR Tools conseguiu apresentar um tempo de processamento inferior. No entanto a ferramenta CPLEX conseguiu expandir a solução alocando mais 2 operadores que permitiram obter um *makespan* final inferior ao apresentado pelo *software* OR Tools. Já na estação WS2_9 a ferramenta OR Tools alocou mais 2 operadores que a ferramenta CPLEX na sua solução para apresentar um *makespan* igual.

Comparando os tempos de processamento para cada estação da linha 1.1 (tabela 11) podemos observar que ambos os *softwares* conseguiram apresentar soluções com valores de *makespan* iguais. Os tempos de execução em cada iteração foram semelhantes.

Observando os tempos que cada ferramenta utilizou para chegar as soluções conseguimos perceber que quantos mais operadores são alocados numa determinada solução, mais tempo necessita o *software* para encontrar a solução. No entanto, para soluções com o mesmo número de operadores, o CPLEX apresenta tempos de execução inferiores. Em soluções com um ou dois operadores, as diferenças de tempo não são muito significativas, mas à medida que a dificuldade dos problemas aumenta, as diferenças também aumentam até ao ponto do OR Tools não conseguir apresentar resultados satisfatórios.

As tabelas 13 e 14 mostram os tempos de permanência de cada peça em cada uma das estações de trabalho.

Tabela 13 – Tempos médios de permanência, OR Tools, cenário 1

Object	Avg Staytime	Min Staytime	Max Staytime
WS1_1	1014.97	420.00	1028.44
WS1_2	1015.21	1015.20	1028.44
WS1_3	0.00	0.00	0.00
WS1_4	0.00	0.00	0.00
WS1_5	40645.97	11734.53	41129.80
WS1_6	41112.28	41112.00	41129.80
WS1_7	13554.13	13554.00	13561.95
WS1_8	0.00	0.00	0.00
WS1_9	0.00	0.00	0.00
WS2_1	42272.32	203.56	42665.70
WS2_2_1	4275.06	4275.00	4281.66
WS2_2_2	42658.27	42658.20	42665.70
WS2_3	34308.08	34308.00	34316.80
WS2_4	4320.00	4320.00	4320.00
WS2_5	4590.11	4590.00	4602.49
WS2_6	3960.00	3960.00	3960.00
WS2_7	15930.15	15930.00	15946.32
WS2_8_1	27091.96	27091.80	27109.75
WS2_8_2	0.00	0.00	0.00
WS2_9_1	3240.18	3240.00	3260.32
WS2_9_2	0.00	0.00	0.00

Tabela 14 – Tempos médios de permanência, CPLEX, cenário 1

Object	Avg Staytime	Min Staytime	Max Staytime
WS1_1	1014.99	420.00	1030.20
WS1_2	1015.22	1015.20	1030.20
WS1_3	0.00	0.00	0.00
WS1_4	0.00	0.00	0.00
WS1_5	40645.72	11730.60	41117.93
WS1_6	41112.09	41112.00	41117.93
WS1_7	4320.13	4320.00	4327.95
WS1_8	0.00	0.00	0.00
WS1_9	0.00	0.00	0.00
WS2_1	15315.63	203.56	15930.00
WS2_2_1	4275.06	4275.00	4281.66
WS2_2_2	6240.67	6240.60	6248.10
WS2_3	15546.61	8640.00	15930.00
WS2_4	15455.72	4320.00	15930.00
WS2_5	15632.00	4590.00	15930.00
WS2_6	15778.90	3960.00	15930.00
WS2_7	15930.15	15930.00	15946.32
WS2_8_1	23130.33	23130.00	23147.95
WS2_8_2	23130.34	23130.00	23148.56
WS2_9_1	3240.18	3240.00	3260.32
WS2_9_2	0.00	0.00	0.00

Para a linha 1.1 os tempos obtidos com cada um dos *softwares* é semelhante. Podemos observar como o tempo de permanência médio na estação WS1_1 (1014.97 s = 16.92 min) é muito superior ao tempo de processamento nessa estação. Isto deve se ao facto de não existirem *buffers* na linha de produção, o que provoca um bloqueio da estação até que a estação seguinte termine o seu processamento permitindo então receber outra peça. O mesmo acontece nas linhas 1.2 e 2.

Já na linha 2 podemos observar como o *bottleneck* (estação mais lenta) da linha, na solução do OR Tools, é a estação WS2_2_2, com um tempo de permanência de 710.97 minutos (42658.27 segundos). Isto provoca que mesmo que as estações de trabalho seguintes tenham capacidade para processar peças em menos tempo, o tempo de ciclo da linha vai ser definido por esta estação. Na solução do CPLEX, para a linha 2 a estação *bottleneck* é a estação WS2_7. No entanto a diferença de tempos não é tão pronunciada como no caso anterior.

Devido ao tempo de ciclo imposto pelas estações mais lentas de cada linha, é possível observar como as máquinas WS2_8_2 (só na solução OR Tools) e WS2_9_2 não são utilizadas. Isto deve se ao facto de as estações paralelas conseguirem realizar todas as tarefas num intervalo de tempo inferior ao do tempo de ciclo imposto pela estação do *bottleneck*.

Observando a ocupação das máquinas (tabela 15) podemos ver várias diferenças em cada uma das linhas de produção, sendo que as maiores diferenças aparecem na linha 2.

Tabela 15 - Ocupação das máquinas, cenário 1

Máquina	*Ocupação - processing OR Tools*	*Ocupação - processing CPLEX*
WS1_1	22.61 %	41.36 %
WS1_2	54.66 %	99.98 %
WS1_3	0 %	0 %
WS1_4	0 %	0 %
WS1_5	15.44 %	28.24 %
WS1_6	54.12 %	98.98 %
WS1_7	17.84 %	10.40 %
WS1_8	0 %	0 %
WS1_9	0 %	0 %
WS2_1	0.46 %	0.83 %
WS2_2_1	9.83 %	17.97 %
WS2_2_2	98.04 %	26.23 %
WS2_3	78.85 %	36.32 %
WS2_4	9.93 %	18.16 %
WS2_5	10.55 %	19.30 %
WS2_6	9.10 %	16.65 %
WS2_7	36.61 %	66.97 %
WS2_8_1	62.27 %	48.62 %
WS2_8_2	0 %	48.62 %
WS2_9_1	7.45 %	13.62 %
WS2_9_2	0 %	0 %

Enquanto que na solução proposta pela ferramenta CPLEX, nenhuma estação apresenta uma ocupação superior a 70%, na solução proposta pela ferramenta OR Tools a estação WS2_2_2 apresenta uma ocupação de 98,04 %.

O mesmo acontece nas linhas 1.1 e 1.2, para as estações WS1_2 e WS1_6 na solução proposta pela ferramenta CPLEX.

Observando a ocupação dos operadores presentes na planta de produção (figuras 22 e 23) podemos ver como a solução proposta pela ferramenta OR Tools produz um desequilíbrio na distribuição do trabalho pelos operadores.

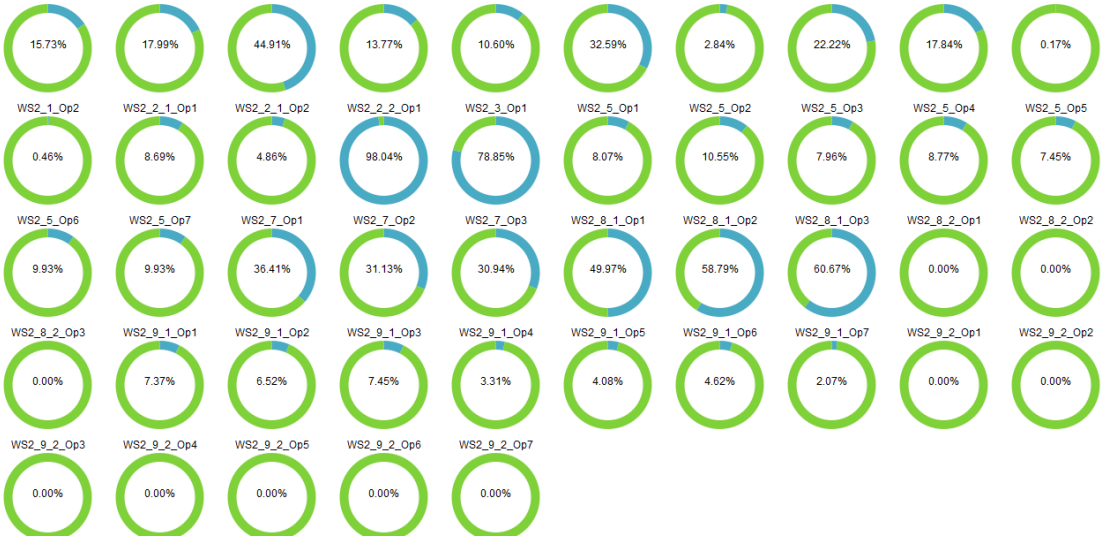

Figura 22 - Ocupação dos trabalhadores, OR Tools, cenário 1

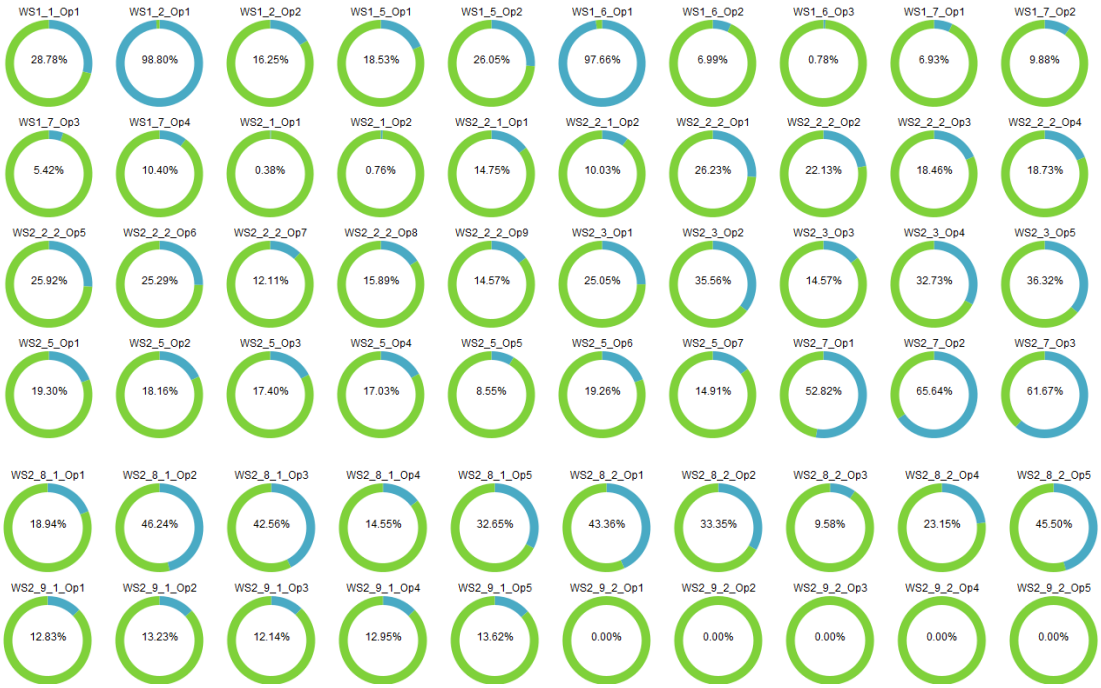

Figura 23 - Ocupação dos trabalhadores, CPLEX, cenário 1

É normal que os operadores alocados às estações de trabalho que não processam nenhuma peça, tenham percentagens de tempo de ocupação nulos.

Podemos observar na tabela 16 a quantidade de operadores que não realizam nenhum trabalho, em qualquer das duas soluções analisadas.

Tabela 16 - Comparação da ocupação dos operadores, cenário 1

Ocupação	*OR Tools*	*CPLEX*
60% - 100%	3	4
30% - 60%	7	10
0% - 30 %	25	46
Sem trabalho	10	5

Por último, a planta de produção que implementa a solução obtida pelo CPLEX conseguiu terminar o plano de produção em 30.28 dias. Já a solução obtida pela ferramenta OR Tools apenas conseguiu terminar o plano de produção em 55.39 dias, o que mostra uma grande diferença no tempo final de execução do plano de produção.

2. Cenário 2 – Cumprir o tempo de ciclo

No cenário anterior foram testadas as capacidades máximas de cada uma das ferramentas de otimização e quais os efeitos práticos na planta de produção. No entanto quando analisamos um problema real não existe uma capacidade ilimitada de recursos humanos. Num problema real existe sempre um compromisso entre tempo de ciclo e recursos utilizados. Quantos mais operadores forem alocados a cada estação menor será o tempo de ciclo da linha de produção, mas os custos de produção serão mais elevados devido aos trabalhadores envolvidos no processo produtivo. Por outro lado, se não alocarmos operadores suficientes corremos o risco de não cumprir com as datas previstas de produção.

2.1 Descrição do cenário de análise

Sendo assim, no cenário 2 o objetivo do nosso problema passa por conseguir cumprir todas as ordens de produção que nos foram propostas dentro do tempo estimado de produção. Como foi indicado anteriormente o nosso plano de produção inclui 2577 peças do tipo 1, 63 peças do tipo 2 e 110 peças do tipo 3. O nosso objetivo será cumprir com as ordens de produção num tempo máximo de 1 mês = 31 dias de trabalho.

Considerando o tempo disponível para completar todas as ordens de produção foi calculado o tempo de ciclo máximo para cada linha utilizando a seguinte fórmula:

$$Tempo\ de\ ciclo\ esperado\ por\ linha = \frac{Tempo\ disponível}{Ordens\ de\ produção na\ linha} \quad (26)$$

O tempo disponível é o mesmo para todas as linhas: 31 dias x 24 horas x 60 min = 44640 minutos.

Na tabela 17 estão apresentados os tempos de ciclo máximo para cada linha de produção.

Tabela 17 – Tempos de ciclo máximos

Linha	*Tempo de ciclo (min)*
Linha 1.1	17.33
Linha 1.2	708.58
Linha 2	405.82

2.2 Resultados da otimização

Uma vez que sabemos quais os tempos de ciclo máximos para cada linha, devemos alocar em cada estação de trabalho o número mínimo de operadores que permita garantir que esses tempos são cumpridos.

Tendo em conta as capacidades de cada uma das ferramentas de otimização foi escolhido o número de operadores a alocar em cada estação, e foi corrido o programa de otimização no modo 1 de forma a obter a alocação das tarefas a cada operador. O número de operadores foi escolhido utilizando os valores de referência que foram criados pelo programa de otimização quando foi utilizado o modo 2 de funcionamento.

Tabela 18 – Alocação e tempos de ciclo, cenário 2

	OR Tools		*CPLEX*	
Máquina	Operadores	*makespan*	Operadores	*makespan*
WS1_1	1	7	1	7
WS1_2	2	16.92	2	16.92
WS1_3	-	-	-	-
WS1_4	-	-	-	-
WS1_5	1	308.58	1	308.58
WS1_6	2	685.2	2	689.7
WS1_7	1	225.9	1	225.9
WS1_8	-	-	-	-
WS1_9	-	-	-	-
WS2_1	1	4.5	1	4.5
WS2_2_1	1	98.22	1	98.22
WS2_2_2	1	710.97	3	245.76
WS2_3	1	571.8	2	286.05
WS2_4	-	72	-	72
WS2_5	2	295.95	2	227.25
WS2_6	-	66	-	66
WS2_7	3	265.5	2	357.3
WS2_8_1	3	451.53	4	388.53
WS2_9_1	1	256.77	1	256.77

Numa análise prévia podemos observar como a otimização feita pela ferramenta OR Tools, à partida, não vai conseguir cumprir o tempo de ciclo máximo pois nas máquinas WS2_2_2 e WS2_3 o menor tempo encontrado é sempre superior ao tempo de ciclo máximo definido.

No caso da estação WS2_8_1, apesar do tempo de ciclo ser superior ao tempo de ciclo máximo definido para essa linha, como esta estação apresenta uma máquina paralela, o tempo de ciclo da estação é reduzido para metade. Sendo assim passamos de 451.53 minutos para 225.77 minutos, que já é inferior que o tempo de ciclo estipulado para a linha 2.

2.3 Validação dos resultados no modelo de simulação

Como foi feito no cenário anterior, de forma a poder validar o modelo de simulação implementado devemos correr a simulação com as soluções propostas no modelo de otimização, e verificar se os tempos de processamento em cada estação correspondem com os tempos que foram obtidos de tais otimizações. Na tabela 19 estão apresentados os tempos de processamento estimados e os tempos da simulação, para o cenário 2 de análise.

Tabela 19 - Tempos de processamento, cenário 2

	OR Tools		*CPLEX*	
Máquina	Tempo desejado (min)	Tempo de simulação (min)	Tempo desejado (min)	Tempo de simulação (min)
WS1_1	7	7	7	7
WS1_2	16.92	16.92	16.92	16.92
WS1_3	0	0	-	0
WS1_4	0	0	-	0
WS1_5	308.58	308.58	308.58	308.58
WS1_6	685.2	686.75	689.7	688.95
WS1_7	225.9	225.9	225.9	225.9
WS1_8	0	0	-	0
WS1_9	0	0	-	0
WS2_1	4.5	4.5	4.5	4.5
WS2_2_1	98.22	99.45	98.22	98.44
WS2_2_2	710.97	710.97	245.76	246.39
WS2_3	571.8	571.8	286.05	285.9
WS2_4	72	72	72	72
WS2_5	295.95	292.35	227.25	227.25
WS2_6	66	66	66	66
WS2_7	265.5	265.5	357.3	357.3
WS2_8_1	451.53	451.53	388.53	388.38
WS2_8_2	451.53	0	388.53	388.38
WS2_9_1	256.77	256.79	256.77	257.07
WS2_9_2	256.77	0	256.77	0

Podemos ver como quase todos os tempos de processamento obtidos na simulação coincidem com os valores esperados. Da mesma forma que no cenário de análise 1, existem alguns casos onde existe uma pequena diferença entre o valor esperado e o valor obtido na simulação, mas que podemos considerar pouco significativas. Isto permite nos validar o modelo de simulação implementado.

2.4 Comparação dos resultados

Em primeiro lugar vamos analisar os tempos de permanência em cada estação (tabela 20 e 21).

Tabela 20 - Tempos de permanência, OR Tools, cenário 2

Object	Avg Staytime	Min Staytime	Max Staytime
WS1_1	1014.97	420.00	1015.20
WS1_2	1015.20	1015.20	1015.20
WS1_3	0.00	0.00	0.00
WS1_4	0.00	0.00	0.00
WS1_5	40841.89	18514.80	41202.00
WS1_6	41202.00	41202.00	41202.00
WS1_7	13554.00	13554.00	13554.00
WS1_8	0.00	0.00	0.00
WS1_9	0.00	0.00	0.00
WS2_1	42272.85	270.00	42658.20
WS2_2_1	5893.20	5893.20	5893.20
WS2_2_2	42658.20	42658.20	42658.20
WS2_3	34308.00	34308.00	34308.00
WS2_4	4320.00	4320.00	4320.00
WS2_5	17541.00	17541.00	17541.00
WS2_6	3960.00	3960.00	3960.00
WS2_7	15930.00	15930.00	15930.00
WS2_8_1	27091.80	27091.80	27091.80
WS2_8_2	0.00	0.00	0.00
WS2_9_1	15406.20	15406.20	15406.20
WS2_9_2	0.00	0.00	0.00

Tabela 21 - Tempos de permanência, CPLEX, cenário 2

Object	Avg Staytime	Min Staytime	Max Staytime
WS1_1	1015.29	420.00	1045.20
WS1_2	1015.52	1015.20	1045.20
WS1_3	0.00	0.00	0.00
WS1_4	0.00	0.00	0.00
WS1_5	40976.06	18518.73	41404.27
WS1_6	41338.26	41337.00	41404.27
WS1_7	13554.13	13554.00	13561.95
WS1_8	0.00	0.00	0.00
WS1_9	0.00	0.00	0.00
WS2_1	20760.79	275.56	21438.00
WS2_2_1	5893.26	5893.20	5899.86
WS2_2_2	14745.67	14745.60	14753.10
WS2_3	21013.95	17154.00	21438.00
WS2_4	20131.92	4320.00	21438.00
WS2_5	21048.59	13635.00	21438.00
WS2_6	21034.05	3960.00	21438.00
WS2_7	21438.15	21438.00	21454.32
WS2_8_1	23303.13	23302.80	23320.75
WS2_8_2	23303.14	23302.80	23321.36
WS2_9_1	15406.38	15406.20	15426.52
WS2_9_2	0.00	0.00	0.00

Podemos observar como em comparação com o cenário 1, os tempos de permanência para a solução do *software* OR Tools não diferem muito. Isto deve se ao facto de as estações mais lentas não terem conseguido alterar os seus tempos de processamento, provocando que todas as máquinas anteriores aumentem os tempos de permanência devido ao bloqueio das máquinas. Para a solução do CPLEX, os tempos apresentam se mais equilibrados.

As estações WS2_8_2 e WS2_9_2, na solução do OR Tools, e a estação WS2_9_2 não solução do CPLEX, não apresentam tempos de permanência, pelo que podemos garantir que não processaram nenhuma peça durante toda a simulação.

Passando à ocupação das máquinas (tabela 22) podemos observar como a distribuição das tarefas pelas estações de trabalho se encontra mais equilibrada na solução da ferramenta CPLEX do que na solução da ferramenta OR Tools. Como foi mencionado anteriormente, a impossibilidade de encontrar uma solução para a estação WS2_2_2 com mais operadores provoca que esta seja a estação *bottleneck* na linha de produção 2. Isto provoca que a estações seguintes apresentem uma taxa de produção inferior a aquilo que seriam capazes. Na solução do CPLEX, ao diminuir o número de operadores em cada estação estamos a fazer que seja necessário mais tempo para completar o processamento de cada peça, pelo que estamos a aumentar a ocupação das máquinas em cada linha de produção.

Tabela 22 - Ocupação das máquinas, cenário 2

Máquina	*Ocupação – processing OR Tools*	*Ocupação – processing CPLEX*
WS1_1	22.50 %	41.05 %
WS1_2	54.38 %	99.26 %
WS1_3	0 %	0 %
WS1_4	0 %	0 %
WS1_5	24.24 %	44.24 %
WS1_6	53.95 %	98.78 %
WS1_7	17.75 %	32.39 %
WS1_8	0 %	0 %
WS1_9	0 %	0 %
WS2_1	0.62 %	1.13 %
WS2_2_1	13.47 %	24.59 %
WS2_2_2	97.53 %	61.52 %
WS2_3	78.44 %	71.57 %
WS2_4	9.88 %	18.02 %
WS2_5	40.10 %	56.89 %
WS2_6	9.05 %	16.52 %
WS2_7	36.42 %	89.45 %
WS2_8_1	61.94 %	48.61 %
WS2_8_2	0 %	48.61 %
WS2_9_1	35.22 %	63.15 %
WS2_9_2	0 %	0 %

Observando a ocupação dos operadores alocados a cada estação (figura 24 e 25) podemos ver um aumento da taxa de ocupação em relação ao cenário 1, para a solução da ferramenta CPLEX. No entanto continuam a existir operadores com taxas de ocupação nulas, nomeadamente aqueles operadores que foram alocados às estações que acabaram por não ter nenhum tempo de processamento.

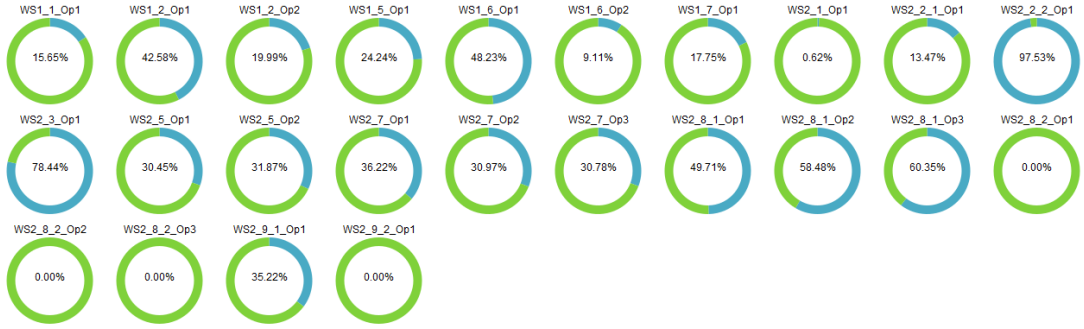

Figura 24 - Ocupação dos operadores, OR Tools, cenário 2

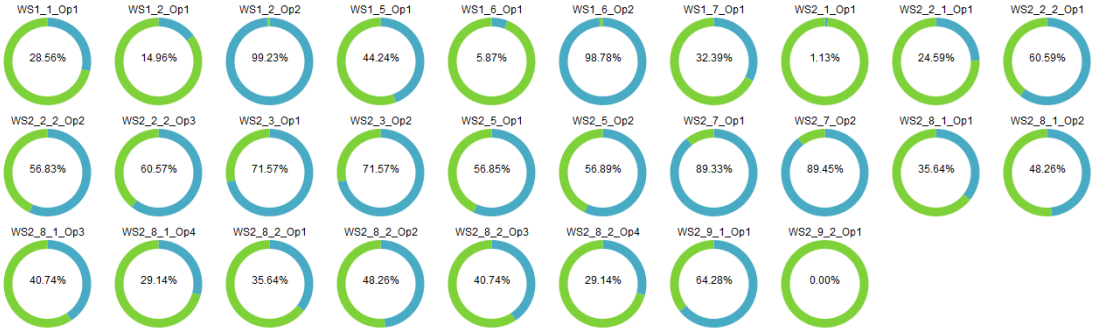

Figura 25 - Ocupação dos operadores, CPLEX, cenário 2

Tabela 23 - Comparação da ocupação dos operadores, cenário 2

Ocupação	*OR Tools*	*CPLEX*
60% - 100%	3	9
30% - 60%	10	11
0% - 30 %	11	8
Sem trabalho	4	1

O tempo total utilizado pela planta de produção para terminar todas as ordens de produção, quando é utilizada a solução do OR Tools, foi de 55.39 dias, pelo que não foi possível cumprir o prazo estimado de produção. Já para a solução do CPLEX, o tempo total necessário para terminar o plano de produção foi de 30.51 dias cumprindo com o tempo de produção estabelecido.

No caso do OR Tools, quando foi feita a otimização sem limite de operadores (cenário 1) foi possível observar como a incapacidade de obter uma solução com mais do que um operador para várias estações limita os tempos de ciclo da linha de produção. Mais concretamente, os prazos de produção do cenário 2 não são cumpridos devido à linha 2. Para que esta ferramenta fosse útil, o tempo de ciclo da linha 2 deveria ser superior ao maior tempo de ciclo da pior estação da linha, a estação WS2_2_2. Considerando o plano de produção utilizado, o tempo total de produção deveria ser, no mínimo, de 55 dias (54.31 dias).

Já no caso do CPLEX é possível obter soluções para planos de produção com horizontes temporais mais apertados. Nesta situação, o tempo de produção mínimo seria de 31 dias (30.21 dias).

Os tempos de ciclo finais apresentados por cada linha de produção segundo a ferramenta de otimização utilizada, em comparação com os tempos de ciclo calculados teoricamente aparecem representados na tabela 24:

Tabela 24 – Tempos de ciclo teóricos vs reais, cenário 2

Linha	*Tempo de ciclo desejado (min)*	*Tempo de ciclo real OR Tools (min)*	*Tempo de ciclo real CPLEX (min)*
Linha 1.1	17.33	31.11	17.05
Linha 1.2	708.58	1272.81	697.45
Linha 2	405.82	728.97	399.45

Os valores dos tempos de ciclo reais foram obtidos utilizando a seguinte fórmula:

$$Tempo\ de\ ciclo = \frac{Tempo\ total\ de\ simulação}{Número\ de\ peças\ produzidas} \quad (27)$$

Os tempos de ciclo reais resultantes da otimização feita pela ferramenta OR Tools, nas linhas 1.1 e 1.2 não representam um valor correto do tempo de ciclo dessas linhas uma vez que a simulação demorou mais tempo a terminar devido ao atraso imposto pela linha 2. É de esperar que os tempos de ciclo das linhas 1.1 e 1.2 sejam semelhantes no caso de uma otimização feita utilizando a ferramenta CPLEX ou OR Tools.

Como foi mencionado o CPLEX conseguiu apresentar uma solução que cumpre com o prazo de produção proposto. No entanto foi possível observar que nesta solução existem duas situações problemáticas:

1 A estação WS2_9_2 tem alocado um operador que não realizou nenhum trabalho.

2 Existem várias máquinas que apresentam tempos de bloqueio consideráveis. Isto gera perdas nas linhas de produção pelo facto da máquina e os operadores associados terem de parar a produção quando não conseguem enviar as peças para as estações seguintes.

Para resolver o primeiro problema devemos fazer uma análise aos maiores tempos de ciclo presentes na linha de produção em estudo, neste caso a linha 2. O nosso objetivo é provar que a estação WS2_9_2 nunca vai ser utilizada de forma a poder descartar a alocação de operadores a esta máquina uma vez que nunca vão realizar nenhum trabalho.

Nesta situação, a estação que apresenta um maior tempo de ciclo é a máquina WS2_7, com um tempo de ciclo de 357.30 minutos (21438.15 segundos) que é menor que o maior tempo de ciclo possível da estação WS2_9_1 (1 operador = 256.77 minutos). Sendo assim, no cenário 2 nunca será necessário alocar operadores à estação WS2_9_2.

Para resolver o segundo problema podemos seguir uma de duas abordagens. Por um lado, podemos alocar menos operadores às estações em questão, mas corremos o risco de não cumprir os tempos de ciclo objetivo.

Por outro lado, podemos aumentar os operadores nas estações seguintes, de forma a que estas trabalhem mais rápido. No entanto com esta solução estamos a aumentar os gastos da linha de produção devido aos operadores extra.

Comparando as duas ferramentas podemos concluir que o CPLEX apresentou um desempenho superior, uma vez que conseguiu cumprir os tempos de produção definidos, o que não aconteceu com a ferramenta OR Tools.

3. Apoio à tomada de decisão

Como foi possível observar, em função do número de operadores que alocamos na nossa planta de produção podemos obter tempos de produção muito variados. Isto faz com que seja difícil tomar uma decisão sobre os operadores e tempos de ciclo na planta de produção.

Com o objetivo de tornar essa tomada de decisão mais simples surgiu a possibilidade de criar um modelo gráfico que apresente as diferentes combinações possíveis permitindo ao utilizador escolher a opção que mais lhe agrade.

Para isso foram criados uns gráficos que apresentam o número mínimo de operadores que uma linha de produção pode alocar, e o tempo de ciclo mínimo esperado para essa linha, com esse número de operadores.

Na figura 26 podemos observar o gráfico do tempo de ciclo em função do número de operadores alocados na linha 1.1, para as duas ferramentas de otimização.

Figura 26 – Modelo de apoio à decisão, linha 1.1

Podemos observar como as duas retas são coincidentes, o que nos permite concluir que para a linha 1.1 as duas ferramentas apresentam o mesmo tipo de opções. No gráfico, ainda que seja representada uma reta apenas devemos considerar os pontos onde o número de operadores (eixo x) seja um número inteiro, uma vez que não faz sentido alocar um operador e meio.

Na figura 27 podemos observar o gráfico do tempo de ciclo em função do número de operadores alocados na linha 1.2, para as duas ferramentas de otimização.

Figura 27 – Modelo de apoio à decisão, linha 1.2

Nesta situação também obtivemos gráficos com as linhas semelhantes pelo que as escolhas oferecidas por cada ferramenta de otimização, para esta linha de produção, são iguais.

Na figura 28 podemos observar os dois gráficos do tempo de ciclo em função do número de operadores alocados na linha 2, para cada uma das ferramentas de otimização.

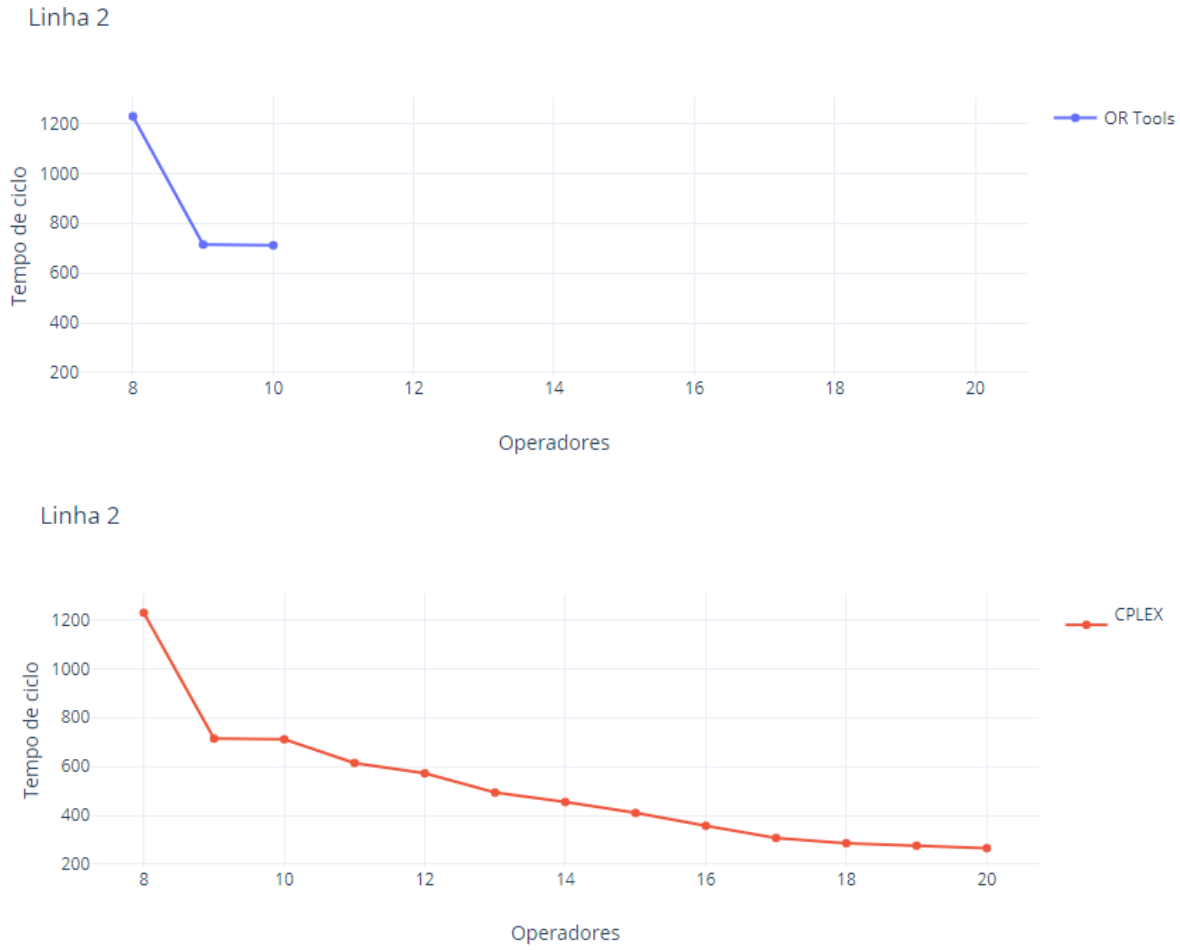

Figura 28 – Modelo de apoio à decisão, linha 2

É possível observar como na linha 2 existe uma grande diferença em função da ferramenta de otimização que é utilizada. Quando é feito o gráfico do tempo de ciclo em função do número de operadores alocados com a ferramenta OR Tools conseguimos obter 3 opções de escolha diferentes. Estas 3 escolhas permitem nos escolher entre 8 a 10 operadores e garantem um tempo de ciclo entre 1228.59 minutos até 710.97 minutos.

Por outro lado, utilizando a ferramenta CPLEX é possível obter um gráfico que apresenta 13 opções diferentes de alocações de operadores. Podemos escolher entre 8 e 20 operadores alocados na linha de produção que garantem um tempo de ciclo entre 1228.59 minutos até 265.5 minutos.

Nesta situação é facilmente percetível como a utilização de uma ferramenta de otimização mais poderosa nos permite tem uma gama de opções muito mais variada.

Capítulo 7 – Conclusões e Trabalho futuro

1. Conclusões

Como referido no início da dissertação a principal dificuldade do nosso caso de estudo consistia na dificuldade em definir e obter os KPIs do processo produtivo. Esta questão originava uma dificuldade adicional na hora de efetuar um correto balanceamento das linhas de produção que se traduz numa falta de produtividade nas linhas de produção. Posto isto, o objetivo que foi definido passava por fazer uma correta alocação dos recursos de forma a melhorar o processo produtivo do caso em estudo.

De forma a cumprir este objetivo foram definidas várias etapas de trabalho, que permitiram estruturar a criação de uma solução válida para o nosso problema.

Em primeiro lugar foi criado um modelo de simulação que representasse as diferentes linhas de produção na fábrica em estudo. Este modelo desenvolvido teve como objetivo reproduzir da forma mais fiel possível todos os aspetos reais do caso em estudo. Por outro lado, também permitiu fazer uma obtenção dos KPIs mais importantes do processo de forma simples, recorrendo à criação de *dashboards* no modelo. Com estes *dashboards* também foi possível validar o funcionamento do modelo na implementação das tarefas realizadas pelos operadores alocados a cada estação de trabalho.

Tendo em conta que os problemas de balanceamento de linhas de produção são comuns nos ambientes fabris, o modelo de simulação que foi criado, com a lógica das tarefas a realizar em cada estação de trabalho pode ser reutilizado noutros projetos que envolvam simulação de linhas de produção.

Foi também desenvolvido um modelo de otimização que permitisse fazer a alocação dos operadores às tarefas a realizar em cada estação de trabalho. Este modelo foi implementado num programa que foi desenvolvido utilizando a linguagem *Python*. O modelo de otimização foi implementado no programa utilizando duas ferramentas de otimização diferentes: CPLEX e OR Tools.

Uma vez que foram utilizadas duas ferramentas de otimização diferentes, todas as soluções obtidas foram comparadas como objetivo de avaliar o desempenho de cada um dos *softwares*.

Uma vez que foram recolhidos todos os dados da otimização, de cada uma das ferramentas utilizadas, recorremos ao modelo de simulação criado anteriormente para validar cada uma das soluções. Foram testados dois cenários de simulação diferentes, um onde não era apresentada nenhuma limitação de operadores que podiam ser alocados na fábrica, e outra onde o objetivo era terminar um plano de produção num determinado espaço temporal.

Relativamente aos KPIs do processo produtivo foi possível observar como existe uma desigualdade no que se refere à distribuição de tarefas por cada estação. Existem estações que realizam muitas tarefas e com regras de precedências complexas entre elas que provocam tempos de ciclo elevados. No entanto outras estações apresentam poucas tarefas e com poucas precedências que produzem tempos de ciclo baixos. Uma vez que estas desigualdades aparecem

ao longo das mesmas linhas de produção, fica muito difícil fazer uma alocação dos operadores de forma a que os tempos de ciclo de cada estação sejam semelhantes.

Uma sugestão de melhoria no caso de estudo passa por fazer uma distribuição das tarefas pelas estações de cada linha de forma mais equilibrada, procurando que os tempos de ciclo de cada estação fiquem mais igualados.

Também foi possível observar como o desempenho do *software* CPLEX apresentou melhores resultados do que o *software* OR Tools. Quando as estações a otimizar apresentavam poucos operadores ou poucas regras de precedência ambas as ferramentas conseguiam resultados satisfatórios, mas noutras situações a falta de poder computacional da ferramenta OR Tools não permitia apresentar soluções válidas.

Foi possível concluir que para casos de estudo onde o número de tarefas a realizar em cada estação, ou o número de precedências, não seja muito elevado a ferramenta OR Tools pode ser uma opção simples e barata para melhorar a produtividade das linhas de produção.

2. Trabalho futuro

Como foi referido anteriormente a ferramenta OR Tools pode ser uma opção simples para melhorar a produtividade de linhas de produção com um grau de complexidade não muito elevado. No entanto, para problemas mais complexos foi possível observar a dificuldade da ferramenta em apresentar resultados satisfatórios.

Considerando que a ferramenta CPLEX requer um investimento elevado por parte do utilizador, uma opção interessante poderia ser o desenvolvimento de uma heurística que permitisse fazer um arranjo inicial das alocações dos operadores às tarefas. Este arranjo inicial iria permitir uma diminuição da complexidade da linha de produção, onde poderia então ser utilizada a ferramenta OR Tools para completar a otimização da linha de produção.

Outra linha de trabalho futuro passaria pela alocação dinâmica dos operadores às estações de trabalho. No nosso caso de estudo o trabalho feito por cada operador não é especializado. Isto significa que qualquer operador pode realizar qualquer tarefa em cada uma das estações presentes na fábrica. Sendo assim, sempre que um operador termina as operações para as quais foi destacado, deveria avaliar o estado das seguintes estações na própria linha de produção, ou até noutras linhas de produção de forma a poder acelerar o processo produtivo. Desta forma é previsível que com o mesmo número de operadores que foram definidos nesta dissertação os tempos de ciclo de cada linha possam diminuir ainda mais.

Por último deveria ser feito um estudo do impacto de tempos de produção estocásticos nos tempos de ciclo de cada linha e analisar a possibilidade do modelo de otimização poder considerar tempos de paragem de produção para descanso dos operadores, ou mesmo para ações de manutenção.

Referências

[1] C. Becker and A. Scholl, “A survey on problems and methods in generalized assembly line balancing,” *Eur. J. Oper. Res.*, vol. 168, no. 3, pp. 694–715, 2006.

[2] A. Almeida, S. Aires, I. M. Noronha, and R. Horta, “Estudo da performance de produção numa linha de enchimento de aerossóis,” 2016.

[3] R. G. Pimenta, “B Alanceamento De L Inhas,” 2011.

[4] M. Fonseca, “Sistema integrado de balanceamento de linhas de produção na indústria do calçado,” 2013.

[5] A. F. V. Fonseca, “Balanceamento de linhas de montagem e aplicação de ferramentas Lean no contexto da Polisport,” 2014.

[6] B. Micieta and V. Stollmann, “Assembly Line Balancing,” *DAAAM Int. Sci. B. 2011*, pp. 257–264, 2011.

[7] N. H. Kamarudin and M. F. F. Ab Rashid, “Modelling of Simple Assembly Line Balancing Problem Type 1 (SALBP-1) with Machine and Worker Constraints,” *J. Phys. Conf. Ser.*, vol. 1049, no. 1, 2018.

[8] S. P. Sari, “No Titleر,” *Pontif. Univ. Catol. del Peru*, vol. 8, no. 33, p. 44, 2014.

[9] A. Bulut, “Simple Assembly Line Balancing Problem (SALBP), Type 2,” pp. 1–9, 2012.

[10] F. C. Pereira, “Modelos integrados de Otimização / Simulação para Balanceamento de Linhas de Produção e Alocação dinâmica de recursos,” 2019.

[11] L. Capacho and R. Pastor, “The ASALB problem with processing alternatives involving different tasks: Definition, formalization and resolution,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3982 LNCS, pp. 554–563, 2006.

[12] J. F. Bard, “Assembly line balancing with parallel workstations and dead time,” *Int. J. Prod. Res.*, vol. 27, no. 6, pp. 1005–1018, 1989.

[13] S. K. Goyal, P. li-Olli, and T. Martikainen, “Equipment selection problems in just-in-time manufacturing systems,” *J. Oper. Res. Soc.*, vol. 44, no. 4, pp. 345–353, 1993.

[14] M. Gen, Y. Tsujimura, and Y. Li, “Fuzzy assembly line balancing using genetic algorithms,” *Comput. Ind. Eng.*, vol. 31, no. 3–4, pp. 631–634, 1996.

[15] R. Pastor, C. Andrés, A. Duran, and M. Pérez, “Tabu search algorithms for an industrial multi-product and multi-objective assembly line balancing problem, with reduction of the task dispersion,” *J. Oper. Res. Soc.*, vol. 53, no. 12, pp. 1317–1323, 2002.

[16] G. Suresh and S. Sahu, “Stochastic assembly line balancing using simulated annealing,” *Int. J. Prod. Res.*, vol. 32, no. 8, pp. 1801–1810, 1994.

[17] S. Hartmann, “A competitive genetic algorithm for resource-constrained project scheduling,” *Nav. Res. Logist.*, vol. 45, no. 7, pp. 733–750, 1998.

[18] V. Biorremedia, D. Tecnologia, H. V. Grupo, E. Centro, U. Federal, and R. Grande, “1. Introdução - Otimização,” *Biologia (Bratisl).*, no. 1, pp. 1–3, 1996.

[19] A. M. Rodrigues, “Sectores e Rotas na Recolha de Res\’{i}duos S\’{\o}lidos Urbanos,” 2014.

[20] M. G. C. Resende, C. C. Ribeiro, F. Glover, and R. Martí, *Scatter Search and Path-Relinking: Fundamentals, Advances, and Applications*. 2010.

[21] S. Hussain and S. J. Khan, “We are IntechOpen , the world ’ s leading publisher of Open Access books Built by scientists , for scientists,” no. November, 2019.

[22] V. H. S. Carneiro, “Abordagens híbridas de Machine Learning / Simulação para sistemas logísticos dinâmicos,” 2019.