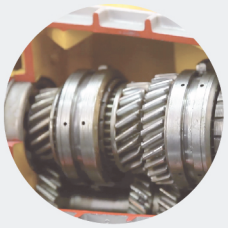
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA E GESTÃO INDUSTRIAL






JOB SEQUENCING AND TOOL SWITCHING PROBLEM

RENATA DA COSTA SANTOS
outubro de 2024

POLITÉCNICO
DO PORTO

ISEP INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

ESCALONAMENTO DE TRABALHOS COM TROCAS DE FERRAMENTAS

Renata da Costa Santos

Dissertação para obtenção do Grau de Mestre em Engenharia e Gestão Industrial

Orientador: Prof. Alzira Maria Teixeira da Mota

Co-orientador: Prof. Luís Adriano Preto Mendes Afonso

Júri:

Presidente:

Prof. Manuel Joaquim Pereira Lopes, Professor Coordenador, Instituto Superior de Engenharia do Porto

Vogais:

Prof. Ana Maria Pinto de Moura, Professora Auxiliar, Universidade de Aveiro

Prof. Alzira Maria Teixeira da Mota, Professora Adjunto, Instituto Superior de Engenharia do Porto

Prof. Luís Adriano Preto Mendes Afonso, Professor Adjunto, Instituto Superior de Engenharia do Porto

Porto, setembro 2024

Resumo

Este trabalho tem como principal objetivo resolver o problema de escalonamento de trabalhos em máquinas paralelas com trocas de ferramentas, considerando ainda *setup*s dependentes da sequência e recursos adicionais. Este estudo surgiu de um problema real, de uma empresa do ramo metalúrgico, que produz ferramentas de alta precisão.

Em primeiro lugar, foi desenvolvido um novo modelo de programação linear inteira mista, com o objetivo de minimizar o *makespan*, isto é, o tempo de conclusão de todos os trabalhos. O modelo foi implementado em *Python* através da biblioteca Pulp e do solver CPLEX. Os resultados demonstraram que este método é uma boa escolha para instâncias de menor dimensão, devido à boa qualidade das soluções e ao baixo tempo computacional, menos de 1 segundo para uma instância com 3 trabalhos e cerca de 2 minutos para uma instância com 14 trabalhos. Contudo, não conseguiu fornecer soluções viáveis para instâncias maiores em tempos computacionais aceitáveis.

Por forma a ultrapassar essa limitação, implementou-se o Simulated Annealing, uma meta-heurística que se diferencia das outras pelo facto de aceitar soluções piores com base em probabilidades. Foram incorporadas outras características do problema que não tinham sido desenvolvidas no modelo matemático, como a restrição de operadores para as etapas de *setup*. Esta meta-heurística também foi implementada em *Python* e foram realizados vários cenários de teste com objetivos diferentes e alguns casos de multiobjectivo. Os resultados da sua implementação demonstraram admissibilidade e estabilidade das soluções, denotando uma boa parametrização realizada. Este método permitiu encontrar soluções para as instâncias de maiores dimensões e em tempo computacional razoável para a indústria. Por exemplo, para uma instância real da indústria metalúrgica com 10 máquinas, 33 trabalhos e 40 ferramentas, o tempo computacional foi aproximadamente de 15 minutos.

Por fim, foi ainda desenvolvida uma heurística construtiva com base numa regra de ordenação dos trabalhos, colocando em primeiro aqueles que requerem mais ferramentas. Esta heurística necessita do input prévio da atribuição dos trabalhos às máquinas. Os resultados refutaram a ideia que colocando primeiramente os trabalhos com mais ferramentas, se evitavam trocas ao longo do processo. Na maioria dos casos, os resultados não foram melhores em relação à solução gerada pela meta-heurística sob as mesmas condições de objetivo de minimização do número de trocas de ferramentas.

Este trabalho fornece algumas contribuições relevantes, nomeadamente a proposta de um novo modelo matemático e a evidência que as meta-heurísticas podem ser muito úteis na resolução deste tipo de problemas de escalonamento em ambiente industrial.

Palavras-chave: Escalonamento, troca de ferramentas, máquinas paralelas, SSP, programação linear inteira mista, Simulated Annealing

Abstract

The main objective of this work is to solve the problem of scheduling jobs on parallel machines with tool switching, considering sequence-dependent setups and additional resources. This study emerged from a real problem in a metalworking company that produces high-precision tools.

Firstly, a new mixed integer linear programming model was developed with the aim of minimizing makespan, meaning the time it takes to complete all the jobs. The model was implemented in Python using the Pulp library and the CPLEX solver. The results showed that this method is a good choice for smaller instances, due to the good quality of the solutions and the low computational time, less than 1 second for an instance with 3 jobs and around 2 minutes for an instance with 14 jobs. However, it failed to provide viable solutions for larger instances in acceptable computational times.

To overcome this limitation, simulated annealing was implemented, a meta-heuristic that differs from others in that it accepts worse solutions based on probabilities. Other characteristics of the problem that had not been developed in the mathematical model were incorporated, such as the restriction of operators for the setup stages. This metaheuristic was also implemented in Python and several test scenarios were carried out with different objectives and some multi-objective cases. The results of its implementation showed that the solutions are feasible and stable, indicating that they have been well parameterized. This method made it possible to find solutions for the largest instances in a reasonable amount of computational time for the industry. For example, for a real instance in the metalworking industry with 10 machines, 33 jobs and 40 tools, the computational time was around 15 minutes.

Finally, a constructive heuristic was developed based on a rule for ordering the jobs, placing those that require the most tools first. This heuristic requires prior input from the assignment of jobs to machines. The results disproved the idea that placing the jobs with the most tools first would avoid trade-offs throughout the process. In most cases, the results were not better than the solution generated by the meta-heuristic under the same objective conditions of minimizing the number of tool changes.

This work provides some relevant contributions, namely the proposal of a new mathematical model and evidence that metaheuristics can be very useful in solving this type of scheduling problem in the industrial environment.

Keywords: Scheduling, tool switching, parallel machines, SSP, mixed integer linear programming, Simulated Annealing

Agradecimentos

A realização desta tese é fruto de uma longa caminhada que não trilhei sozinha. Várias pessoas, direta ou indiretamente, contribuíram para que conseguisse alcançar este objetivo e, por isso, a elas sou profundamente grata.

Em primeiro lugar, agradeço à professora Alzira Mota e minha orientadora, por toda a paciência, apoio e orientação ao longo de todo este processo, e por ter sido uma peça fundamental neste resultado atingido. Agradeço, ainda, ao professor Luís Afonso, meu coorientador, pela sua dedicação e pela partilha de conhecimentos.

Ao professor Manuel Pereira Lopes, sou grata pelos ensinamentos que contribuíram para o meu crescimento académico e profissional.

Agradeço ainda a oportunidade de ter participado na Conferência Metalomecânica da Supply Chain Magazine e no 11º Encontro Nacional de Engenharia e Gestão Industrial com este trabalho.

À minha família, que é a minha base e o meu maior suporte, deixo o meu mais profundo agradecimento. À minha mãe, Teresa, pela educação, valores e apoio incondicional, independentemente das circunstâncias. A minha eterna gratidão por todo o amor e sacrifício ao longo de toda a minha vida. À minha avó materna, Conceição, e aos meus tios e padrinhos, Pedro e Anésia, pelo carinho, pela confiança e por me motivarem a ser sempre melhor.

Aos meus amigos, agradeço todas as trocas de ideias durante os momentos mais desafiadores desta caminhada, a companhia e o incentivo que tornaram este caminho muito mais leve e agradável.

Por fim, a todos que, de alguma forma, contribuíram para que este trabalho se realizasse o melhor possível e para que esta meta fosse alcançada.

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

CNC	Máquinas de controlo numérico computorizado
FMS	*Flexible Manufacturing Systems*
IPMTC	*Identical Parallel Machines problem with tooling constraints*
SSP	*Job Sequencing and Tool Switching Problem*
SSP-NPM	*Job Sequencing and Tool Switching Problem with non-identical parallel machines*
EDD	*Earliest Due Date*
PLIM	Programação Linear Inteira Mista
SA	Simulated annealing
APMX	*Adapted Partial-mapped Crossover*

1 Introdução

Esta dissertação de mestrado pretende resolver um problema de escalonamento de tarefas e de trocas de ferramentas em máquinas paralelas. O trabalho tem por base dados reais de uma empresa da indústria metalúrgica. Para isso, tem-se em consideração a atribuição de trabalhos às máquinas e a sua sequenciação, de maneira a reduzir ao máximo as tarefas de mudança de ferramentas.

Assim, neste capítulo é realizado em primeiro lugar um enquadramento do tema para se entender a sua pertinência. De seguida é exposto o problema de investigação, focando nos objetivos deste projeto. Por fim, é ainda referida a metodologia adotada, bem como as razões que levaram à sua escolha.

1.1 Problema de investigação, enquadramento e pertinência

Nos dias de hoje, a flexibilidade e a adaptabilidade são das características mais importantes para se atingir um sistema produtivo eficaz. Isto deve-se, principalmente à crescente complexidade de requisitos que os compradores exigem e à grande diversidade de produtos existentes. A base dum sistema produtivo são as máquinas, pelo que é relevante fazer uma análise atenta a esta componente (Cura, 2023).

Com a crescente aposta na tecnologia, foram desenvolvidas máquinas que são capazes de desempenhar vários tipos de tarefas, substituindo assim a necessidade de vários equipamentos. Estas máquinas podem fazer operações como torneamento e fresagem, precisando apenas de trocar a ferramenta (Cura, 2023). Este tipo de equipamento é muitas vezes associado ao termo *Flexible Manufacturing System* (FMS), já que este procura acrescentar versatilidade e flexibilidade à produção em grande escala. Um FMS é um sistema flexível de fabrico, que conta com uma grande componente automatizada e, também, com máquinas do mesmo género das referidas (Beezão et al., 2017). Grande parte destas máquinas têm uma espécie de armazém de ferramentas embutido, designado por carrossel, que permite guardar um certo número de ferramentas. Contudo, tal como seria esperado, tem uma capacidade limitada e para fazer face à elevada variedade de tarefas e produtos, torna-se necessária a troca

de ferramentas deste carrossel. Na grande maioria dos sistemas produtivos, esta troca de ferramentas é dos processos mais demorados e, por isso, deve ser meticulosamente estudado, de modo a reduzi-los (Cura, 2023).

Este estudo enquadra-se no *Job Sequencing and Tool Switching Problem* (SSP), que se traduz num problema de escalonamento de trabalhos com a condicionante da troca de ferramentas nas máquinas. O SSP geralmente está associado à implementação de um FMS e pode ser encontrado nas mais diversas indústrias, tais como, metalúrgica, química e até na gestão da memória de computadores. Este problema já é relativamente antigo e altamente estudado, mas ainda há muitas vertentes por analisar (Rifai et al., 2022-b). O problema mais básico, que consiste em apenas uma máquina e que considera que os tempos de troca são uniformes, foi abordado pela primeira vez em (Tang & Denardo, 1988).

Ao longo dos anos, muitos autores abordaram este tema de diversas formas na tentativa de encontrarem a melhor solução para este tipo de problema. No artigo de (Calmels, 2019) é feita uma análise pormenorizada à literatura existente e nele são enumeradas as várias técnicas utilizadas. Começando por modelos matemáticos e avançando pelas heurísticas e meta-heurísticas, foram muitas as técnicas usadas, com especificações distintas. Também é possível encontrar técnicas de multiobjectivo que procuram, nomeadamente, minimizar o número de paragens para troca de ferramentas e o *makespan*.

Contudo, grande parte destes modelos estudados não reflete totalmente a realidade, uma vez que considera que os tempos de trocas são independentes da sequência. Para além disso, a maior parte só conta com uma máquina flexível e isso não é compatível com o mundo real (Calmels, 2019). O artigo de (Mara et al., 2023), que tinha como um dos seus objetivos comparar um SSP básico, com tempos de troca uniformes, e um SSP considerando a sequência para os tempos de mudança de ferramentas, revelou que estes geram soluções bastante diferentes. Ter em consideração que os tempos não são uniformes resultou num tempo total de *setups* superior para os mesmos dados.

Posto isto, há então a necessidade de estudar mais aprofundadamente este tema, considerando outras características reais na resolução deste problema, que até então não foram tidas em conta.

1.2 Questão e objetivos de investigação

De maneira a tentar colmatar as lacunas existentes na literatura referente a este tipo de problema, surge este trabalho que visa o desenvolvimento de uma ferramenta de apoio à decisão para resolver um problema de escalonamento de tarefas e de mudanças de ferramentas em máquinas paralelas. Este problema apresenta características reais de uma empresa do ramo da metalurgia que produz ferramentas e materiais de manutenção. A empresa em questão possui máquinas com capacidade de efetuar vários tipos de trabalhos, tendo apenas a condicionante do número de ferramentas que a máquina consegue armazenar para a produção. O principal objetivo é então criar um algoritmo que faça o escalonamento, ou

seja, que atribua os trabalhos às máquinas e que os sequencie, e que estipule as trocas de ferramentas a realizar em cada instante do processo.

Com este projeto abordam-se questões pouco referidas na literatura, como a consideração de múltiplas máquinas flexíveis e os tempos de *setup* não uniformes. Além disso, há também uma grande condicionante que é o número de operadores disponíveis para efetuar os *setups* às máquinas. Portanto, existem várias restrições e especificações a ter em conta neste caso, o que o torna mais rico e complexo.

Tendo em conta este objetivo geral da investigação, podem ser formulados os seguintes objetivos mais específicos:

- Criação de um modelo matemático, com várias restrições, nomeadamente montagem e desmontagem de ferramentas e alteração do diâmetro da haste;

- Desenvolvimento de uma meta-heurística e de uma heurística construtiva;

- Implementação em linguagem *Python* do modelo matemático (com a biblioteca *Pulp*), e da meta-heurística e da heurística construtiva;

- Avaliação do desempenho de todas as abordagens realizadas, bem como comparação de resultados atingidos entre elas.

1.3 Opções metodológicas

Para o desenvolvimento deste trabalho foi necessária muita pesquisa prévia, para entender o tema e toda a atmosfera envolvente. Este projeto insere-se num tipo de problema, o SSP, para o qual já existem várias abordagens e, por isso, para desenvolver algo novo e diferente é necessário analisar o que já existe. Desta forma, a investigação atuante demonstra ser a metodologia mais correta a utilizar, uma vez que utiliza a teoria para gerar conhecimentos científicos práticos. Este tipo de investigação é comumente usado em situações cujo problema está bem fundamentado e descrito, para o qual é sabido o motivo e quais os fatores condicionantes. Pretende-se a mudança, ou seja, descobrir novas alternativas para resolver problemas já conhecidos (Oliveira, 2011).

Por conseguinte, em primeiro lugar é feita uma pesquisa e análise intensivas sobre tudo acerca do tema em discussão, desde a sua origem e importância, até às várias técnicas utilizadas pelos autores ao longo dos anos. Só depois de analisada a literatura existente sobre este assunto e estudadas todas as vantagens e desvantagens das estratégias já implementadas, é possível selecionar uma ou mais abordagens para a resolução deste problema. Por fim, são desenvolvidos os algoritmos, que são validados e testados em instâncias com dados reais.

1.4 Estrutura do relatório

Depois da Introdução, existem mais cinco capítulos neste trabalho: Revisão Bibliográfica, Modelo de Programação Linear Inteira Mista, Meta-Heurística e Heurística Construtiva, Resultados Computacionais e, por fim, Conclusão.

No capítulo 2, Revisão Bibliográfica, seguiu-se uma lógica do geral para o particular, começando pelos problemas de escalonamento e a importância dos *setups* neste tipo de problemas, até ao problema em estudo, o SSP. Neste capítulo, são ainda analisadas várias abordagens à resolução destes problemas, desde modelos matemáticos a meta-heurísticas.

No capítulo 3, Modelo de Programação Linear Inteira Mista, fornece uma descrição detalhada do problema e as alterações principais feitas face ao modelo usado como referência para a criação do novo modelo. É depois apresentado o modelo de programação linear inteira mista desenvolvido, bem como as limitações que este apresenta.

No capítulo 4, Meta-Heurística e Heurística Construtiva, são explicadas a meta-heurística implementada, o Simulated Annealing e a heurística construtiva desenvolvida. É ainda apresentado o processo de parametrização do Simulated Annealing.

No capítulo 5, Resultados Computacionais, primeiramente são apresentadas algumas características das instâncias de teste geradas. Depois, apresentam-se os resultados obtidos com todos os métodos utilizados e é feita uma comparação entre eles, de maneira a avaliar o desempenho das abordagens implementadas.

No capítulo 6, Conclusão, são tecidas algumas conclusões do estudo realizado, nomeadamente os principais contributos e desafios enfrentados. São, ainda, discutidas algumas perspetivas para trabalho futuro.

2 Revisão Bibliográfica

Com o intuito de entender todo o paradigma por trás deste problema de investigação, este capítulo começa por abordar os problemas de escalonamento e o papel dos *setups* nestes. Depois é analisado mais pormenorizadamente o contexto do problema em estudo e todo o seu *background* existente, através do estudo de várias técnicas já implementadas. Finalmente, é apresentado o modelo matemático de referência utilizado na criação do novo modelo.

2.1 Problemas de escalonamento

O escalonamento de operações é um grande desafio para o planeamento e gestão dos processos de fabrico. Esta tarefa pode ser muito simples ou muito complexa, dependendo das condições do problema, tais como o ambiente, as restrições de recursos e do processo e, ainda, os indicadores de desempenho escolhidos para a sua avaliação (Pezzella et al., 2008).

É possível classificar os problemas de escalonamento com base em inúmeros fatores, como por exemplo, número de trabalhos a executar, número de máquinas disponíveis, tempos/custos de *setup*, entre outros. A maneira mais utilizada de classificação é a notação de três campos α/β/γ. Assim, segundo (Allahverdi, 2015), os três campos são:

- Campo α: utilizado para descrever o ambiente de trabalhos em relação à disposição das máquinas;

- Campo β: refere as outras condições do processo, como por exemplo em relação aos *setups*;

- Campo γ: apresenta a medida de desempenho escolhida para a avaliação da solução.

Na Figura 1, encontram-se descritos estes campos em pormenor, com as várias opções disponíveis.

α		β		γ	
Notation	Description	Notation	Description	Notation	Description
1	Single machine	ST_{si}	Sequence-independent setup time	C_{max}	Makespan
P	Parallel machines (identical)	SC_{sd}	Sequence-dependent setup cost	E_{max}	Maximum earliness
Q	Parallel machines (uniform)	ST_{sd}	Sequence-dependent setup time	L_{max}	Maximum lateness
R	Parallel machines (unrelated)	$ST_{si,f}$	Sequence-independent family setup time	T_{max}	Maximum tardiness
Fm	m-stage flowshop	$ST_{sd,f}$	Sequence-dependent family setup time	D_{max}	Maximum delivery time
FFm	m-stage flexible (hybrid) flowshop	$SC_{sd,f}$	Sequence-dependent family setup cost	TSC	Total setup/changeover cost
AFm	m-stage assembly flowshop	ST_{psd}	Past-sequence-dependent setup time	TST	Total setup/changeover time
J	Job shop	$Prec$	Precedence constraints	TNS	Total number of setups
FJ	Flexible job shop	r_j	Non-zero release date	ΣF_j	Total flowtime
O	Open shop			ΣC_j	Total completion time
				ΣE_j	Total earliness
				ΣT_j	Total tardiness
				ΣW_j	Total waiting time
				ΣU_j	Number of tardy (late) jobs
				$\Sigma w_j C_j$	Total weighted completion time
				$\Sigma w_j F_j$	Total weighted flowtime
				$\Sigma w_j U_j$	Weighted number of tardy jobs
				$\Sigma w_j E_j$	Total weighted earliness
				$\Sigma w_j T_j$	Total weighted tardiness
				$\Sigma w_j W_j$	Total weighted waiting time
				$\Sigma h(E_j)$	Total earliness penalties
				$\Sigma h(T_j)$	Total tardiness penalties
				$TADC$	Total absolute differences in completion times

Figura 1 - Notação de 3 campos - α/β/γ (Allahverdi, 2015)

No caso do problema em estudo, a notação é P/ST_{sd}/C_{max} uma vez que se trata de um problema de escalonamento em máquinas paralelas idênticas (ou seja, todas as máquinas podem realizar todos os trabalhos) com os tempos de setup dependentes da sequência escolhida e cujo objetivo é minimizar o makespan (tempo total de funcionamento das máquinas).

2.2 O papel dos setups

Nos ambientes atuais de fabrico e de serviços, o escalonamento que envolve os tempos/custos de *setup* desempenha um papel essencial. Estes processos de preparação para a realização de trabalhos, não são tarefas de valor acrescentado e, por isso, têm de ser considerados nas decisões de escalonamento. Através da inclusão deste fator nas decisões de planeamento, é expectável um aumento de produtividade, redução de desperdícios, maior eficiência na utilização de recursos e cumprimento dos prazos. O interesse por esta componente industrial começou em meados de 1960 e tem vindo a aumentar ao longo dos anos (Allahverdi, 2015).

De acordo com (Allahverdi & Soroush, 2008), o tempo de *setup* pode ser definido como "o tempo necessário para preparar os recursos (por exemplo, máquinas, pessoas) para executar uma tarefa (por exemplo, trabalho, operação)". Estas atividades de preparação incluem inúmeras possibilidades, como limpeza, inspeção do material, posicionamento do material no processo, ou mesmo colocação de ferramentas e ajustes de parâmetros, como no caso em estudo. Também outras atividades não tão físicas são consideradas *setups*, tais como, transferir programas ou ficheiros e executá-los num sistema informático (Allahverdi & Soroush, 2008).

Segundo um levantamento realizado acerca de problemas de escalonamento, mais de 90% da literatura não considera os tempos/custos de *setups*. Ignorar estes tempos até pode ser válido para algumas aplicações, mas nos restantes casos prejudica muito a qualidade das soluções (Allahverdi, 2015).

É possível reduzir os *setups* de várias formas, através de melhores processos de preparação, de técnicas como o SMED, e também realizando um melhor escalonamento das tarefas, para evitar o número de vezes que a máquina deve parar. As vantagens de reduzir estes tempos encontram-se apresentadas na Figura 2.

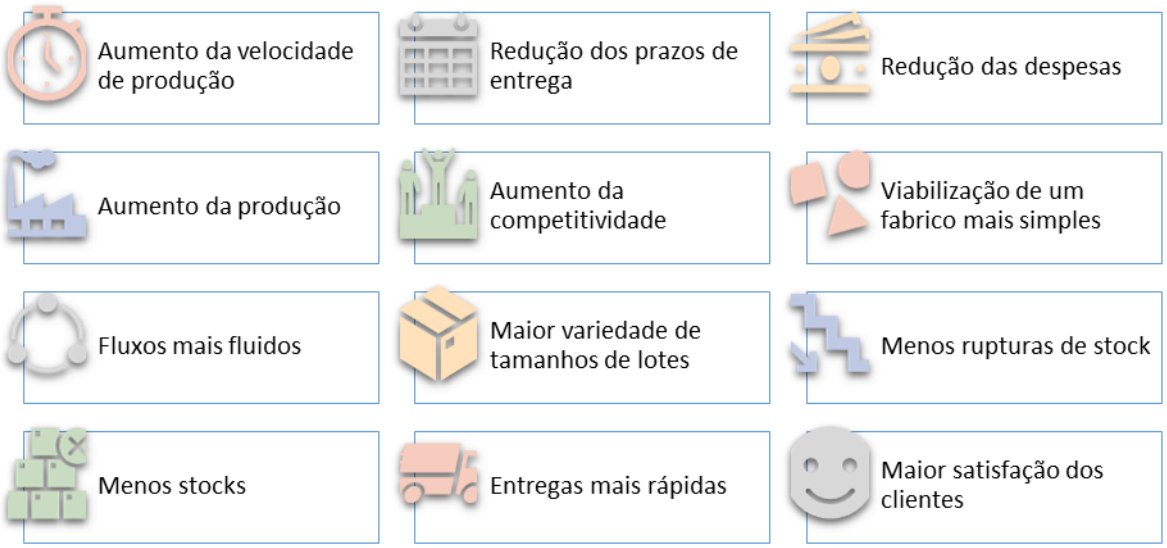


Figura 2 - Vantagens da redução dos setups (adaptado de (Allahverdi & Soroush, 2008))

2.3 Sistemas de fabrico flexíveis

Ao longo dos anos, as empresas de fabrico têm enfrentado muitos desafios, sendo um deles a diferenciação e personalização dos produtos requerida pelo mercado comprador. De forma a superar este compromisso entre as economias de escala e de gama, é fundamental analisar os conceitos de fabrico. Isto também é relevante para conseguir acompanhar a Indústria 4.0 e as fábricas inteligentes. Estas apresentam características que as distinguem muito bem das outras, como a elevada flexibilidade e a automatização da produção. A partir da década de 80, algumas empresas mais focadas na transformação começaram a utilizar os *flexible manufacting systems* para oferecer uma maior gama de produtos aos compradores, tentando colmatar os seus requisitos (Calmels, 2019).

Geralmente, um FMS é composto por máquinas de controlo numérico computorizado (CNC) ligadas entre si e que têm capacidade para realizar vários tipos de trabalhos, necessitando apenas das ferramentas disponíveis. Podem realizar tarefas como perfuração e torneamento, sem ser necessário haver uma alteração repentina entre operações, fazendo com que a produção seja mais dinâmica (Rifai et al., 2022-a).

Cada máquina CNC normalmente é composta por um carrossel de ferramentas (*tool magazine*), que possui um limite de capacidade. Na Figura 3, é possível ver um exemplo de um carrossel de ferramentas de uma máquina CNC.

Figura 3 - Carrossel de ferramentas de uma máquina CNC

Habitualmente, consegue guardar o número de ferramentas necessário para produzir um trabalho até ao fim. Contudo, com o aumento da variedade de produtos e a mudança de paradigma de produção em grande volume para alta customização, o número de ferramentas necessário para a realização das operações aumentou bastante. Portanto, quando se excede a capacidade máxima do carrossel de ferramentas da máquina, passa a ser indispensável realizar trocas, ou seja, tirar algumas existentes para poder colocar novas. Estas trocas continuam a ser um dos maiores desafios, apesar dos avanços da tecnologia terem ajudado muito neste problema. Segundo (Beezão et al., 2017), estas mudanças de ferramentas demoram mais tempo do que qualquer outra atividade de preparação, representando cerca de 25 a 30% do custo total num FMS. Assim, a troca de ferramentas demonstra ter uma grande influência na eficiência destes sistemas (Dang et al., 2021).

Esta decisão de troca de ferramentas está interligada logicamente com o sequenciamento do trabalho. Este problema é conhecido como *Job Sequencing and Tool Switching Problem*, que basicamente se traduz no problema de escalonamento de trabalhos conjugado com a mudança de ferramentas. A metalomecânica é uma das áreas onde este problema normalmente se verifica (Dang et al., 2021).

2.3.1 Problema de escalonamento de trabalhos e trocas de ferramentas

O SSP pode ser descrito como um problema de escalonamento de trabalhos ou tarefas que, por sua vez, necessitam de ferramentas para serem realizados. O SSP básico, que considera apenas uma máquina flexível, foi primeiramente debatido em (Tang & Denardo, 1988). Neste artigo, os autores propunham decompor o problema em duas componentes distintas: o escalonamento dos trabalhos e a troca de ferramentas. Contudo, a partir do básico consegue-se chegar a várias variantes, modificando pressupostos e restrições, tais como, o tamanho das ferramentas e os tempos de trocas (Calmels, 2019).

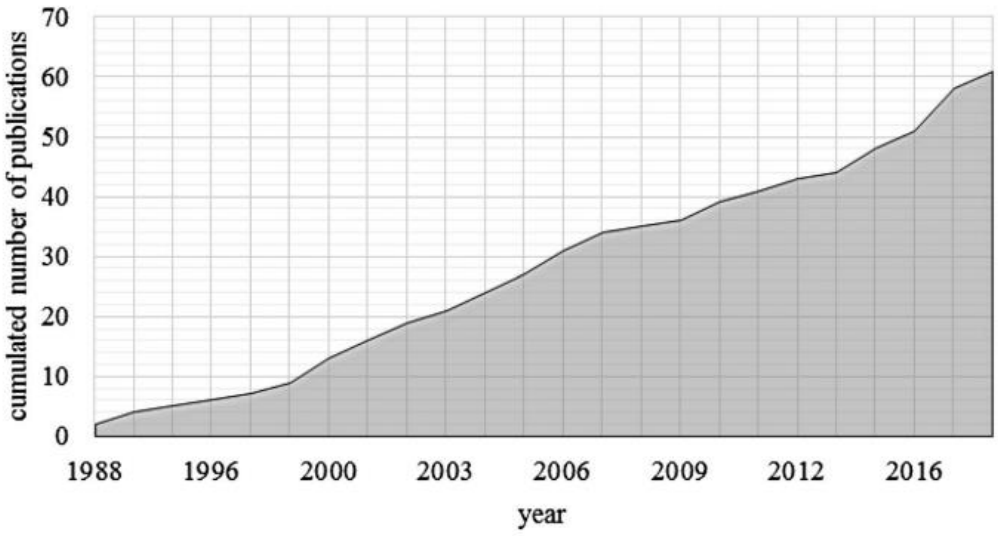


Figura 4 - Publicações sobre SSP ao longo dos anos (Calmels, 2019)

A Figura 4 demonstra que o número de modelos criados relacionados com o SSP ou as suas variantes tem vindo a aumentar gradualmente ao longo dos anos. Apesar de este tipo de problema ser bastante abordado, e com uma vasta revisão, de acordo com (Rifai et al., 2022-a), grande parte da literatura debate apenas o SSP básico. No entanto, os tempos de trocas de ferramentas podem ser uma condicionante quando são diferentes entre si. Isto faz com que a ordem em que são feitas as coisas tenha mais importância e impacto no tempo. A título de exemplo, ao imprimir algo que necessita de sobreposição de cores, é importante saber a sequência que, por sua vez, irá resultar num tempo de troca diferente. O mesmo acontece na indústria metalomecânica, no fabrico de chapas por exemplo (Rifai et al., 2022-a).

Estas trocas podem ser designadas de tempos de *setup*, uma vez que são tarefas não produtivas executadas para preparar as máquinas para os próximos produtos. Estão incluídos nestes tempos operações como limpeza, mudança de ferramentas ou fixação de alguma peça. É de extrema relevância considerá-los, já que desta forma as condições são mais próximas da realidade (Mara et al., 2023).

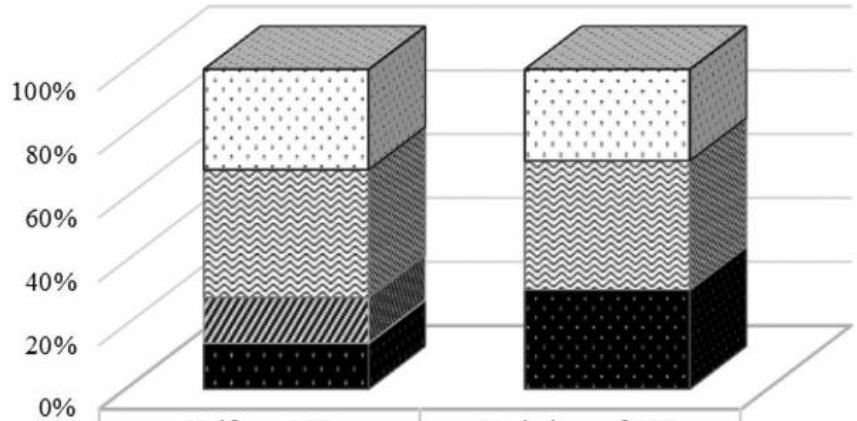
Além disso, considerar apenas uma máquina, como no SSP básico, não é muito realista, nem reflete verdadeiramente um ambiente industrial. Daí surgem outras variantes de problema com múltiplas máquinas paralelas, podendo estas ser idênticas ou não. A versão com máquinas semelhantes é muitas vezes apelidada de *Identical Parallel Machines problem with tooling constraints* (IPMTC). Esta vem sendo estudada desde os anos 80, apesar de os problemas de escalonamento em máquinas paralelas serem debatidos desde os anos 50 (Soares & Carvalho, 2020). Já a variante com máquinas distintas é conhecida *por Job Sequencing and Tool Switching Problem with non-identical parallel machines* (SSP-NPM), que se adapta melhor aos desafios que a indústria enfrenta nos dias de hoje. O serem não idênticas pode significar terem carrosséis de ferramentas com diferentes capacidades e/ou tempos de trocas distintos. Por isso mesmo, a sequência de trabalhos em cada máquina passa a ter um impacto ainda maior nas trocas e nos respetivos tempos (Cura, 2023).

Normalmente, estes problemas têm como objetivo minimizar os tempos de mudança de ferramentas ou o número de trocas, mas existem muitos outros objetivos que podem ser escolhidos em detrimento dos anteriormente mencionados, tais como a minimização do tempo de fluxo total, do atraso total, do *makespan*, entre outros (Cura, 2023).

2.4 Abordagens de resolução

Por forma a tentar dar solução a este tipo de problemas, vários autores desenvolveram modelos, de diversos tipos. Geralmente, para problemas de otimização de escalonamento, tal como o que está a ser discutido nesta dissertação, os métodos mais utilizados são as heurísticas, meta-heurísticas e modelos matemáticos. Comparando a utilização das meta-heurísticas e heurísticas com os modelos matemáticos, percebe-se que estes últimos são menos utilizados. Isto pode estar relacionado com o facto de estes serem muito morosos e, por isso, só serem usados para pequenas instâncias de problemas. As heurísticas construtivas foram as mais usadas, uma vez que demonstram ter um bom desempenho em problemas com maior quantidade de dados (Calmels, 2019).

Na Figura 5, é possível visualizar isso mesmo, uma preferência maior pelas heurísticas face, por exemplo, aos modelos matemáticos, tanto para o SSP básico como para as variações.



	Uniform SSP	Variations of SSP
Metaheuristic	11	12
Heuristic	14	17
Exact	5	0
Math. Model	5	13

Figura 5 - Frequência de soluções por tipo de abordagem (Calmels, 2019)

Laporte et al. (2002) introduziram algoritmos branch-and-cut e branch-and-bound para o SSP, comparando duas formulações de programação linear inteira. Ghiani et al. (2007) desenvolveram um algoritmo branch-and-bound para o problema de carregamento de ferramentas com uma nova formulação simétrica, no âmbito de um centro de maquinação com controlo numérico. Em 2010, Ghiani et al. formularam o SSP como um problema não linear de

ciclo hamiltoniano de menor custo, propondo um algoritmo branch-and-cut e comparando-o com procedimentos exatos anteriores.

Burger et al. (2015) abordaram um problema de escalonamento na indústria de embalagens, considerando trabalhos de impressão multicoloridos com tempos de configuração dependentes da sequência. Paiva et al. (2017) introduziram uma metodologia utilizando representações gráficas, métodos heurísticos e técnicas de pesquisa local para minimizar o número de trocas de ferramentas no SSP.

Ahmadi et al. (2018) discutiram a redução do SSP ao Problema de Sequenciação de Tarefas (JSeP), introduzindo um método híbrido combinando 2-TSP (*Traveling Salesman Problem of Second Order*) e um novo algoritmo genético. Silva et al. (2021) propuseram um novo modelo de fluxo de múltiplas mercadorias para o SSP considerando um limite inferior na relaxação linear baseado no número de ferramentas e na capacidade da máquina. Calmels (2022) apresentou um procedimento iterativo de pesquisa local para o SSP com máquinas paralelas não idênticas, considerando vários objetivos, tais como a minimização do tempo de fluxo total, do *makepan* e do número total de trocas de ferramentas.

Estes são alguns exemplos de abordagens feitas nos últimos anos, na tentativa de resolver o SSP o melhor possível. Contudo, nos próximos tópicos são apresentadas mais detalhadamente algumas perspetivas seguidas por outros autores no passado, agrupadas por categorias, por forma a perceber as características de cada uma e quais as vantagens da sua utilização.

Contudo, comparar os algoritmos é bastante complexo devido à utilização de instâncias de problemas diferentes. Grande parte dos autores usam os seus próprios dados, muitas vezes gerados aleatoriamente, o que faz com que estes pareçam pequenos relativamente às dimensões de um problema real. Isto deve-se, em parte, às taxas de divulgação e aceitação de instâncias de referência sobre problemas SSP serem significativamente baixas. Desta forma, é complicado fazer comparações entre resultados, uma vez que raramente partem da mesma base de dados. A utilização de instâncias de referência seria muito benéfica, pois iria permitir ter uma base de comparação e perceber se os resultados estão a melhorar ou piorar. Para além disso, também seria útil ter problemas com dimensões mais realistas, para espelhar da melhor forma as características das situações para as quais se tenta encontrar solução (Calmels, 2019).

2.4.1 Modelos matemáticos

Apesar de se ter verificado no tópico anterior que os modelos matemáticos não são as abordagens mais escolhidas, muitos autores envergam por este caminho. Um exemplo disso é o artigo de (Beezão et al., 2017) no qual são propostas duas formulações matemáticas para minimizar o *makespan*, num problema SSP em máquinas paralelas idênticas. Ambas são inspiradas em trabalhos anteriores de outros autores e distinguem-se uma da outra pela forma como as restrições foram elaboradas: um dos modelos conta com três variáveis binárias, enquanto o outro conta com cinco. Contudo, ambas as formulações seguem algumas regras tais como cada trabalho só poder ser atribuído a uma única máquina, não poderem existir

interrupções e os tempos de processamento de cada trabalho serem iguais para todas as máquinas. Neste caso, o tempo de troca de ferramentas é considerado constante, ou seja, é independente da sequência de trabalhos.

Ambos os modelos foram testados para um mesmo conjunto de 30 instâncias de problema. O primeiro modelo, com as três variáveis binárias, demonstrou atingir melhores resultados, embora se tenha percebido que para problemas com alguma dimensão, como por exemplo, 15 máquinas, ambos os modelos apresentam dificuldades e tornam-se mais morosos. Portanto, este exemplo vem reforçar a ideia já referida de que este tipo de técnica não é a melhor opção quando se pretende atingir boas soluções em pouco tempo computacional (Beezão et al., 2017).

O clássico problema do caixeiro-viajante (TSP) também pode ser utilizado para resolver problemas deste género, tal como se pode ver em (Mara et al., 2023). Para isso, é criado um trabalho fictício que representa as tarefas de início e de paragem, tal como um depósito no TSP tradicional. Depois, conforme o próprio modelo, cada vértice, que neste caso representa um trabalho, deve ser visitado apenas uma vez.

Nesse mesmo artigo, (Mara et al., 2023) é apresentado mais um modelo matemático, conhecido por *multicommodity flow*, que apresenta num grafo o fluxo de transferência de várias mercadorias. A Figura 6 auxilia na compreensão do funcionamento deste mecanismo. Os quadrados são trabalhos a ser realizados em determinada máquina, neste caso 6, e os círculos são as ferramentas disponíveis. Apesar de existirem neste exemplo 5 ferramentas disponíveis, a máquina só tem espaço no carrossel para 3. Portanto, as ferramentas armazenadas para cada trabalho encontram-se assinaladas pelos círculos a sombreado.

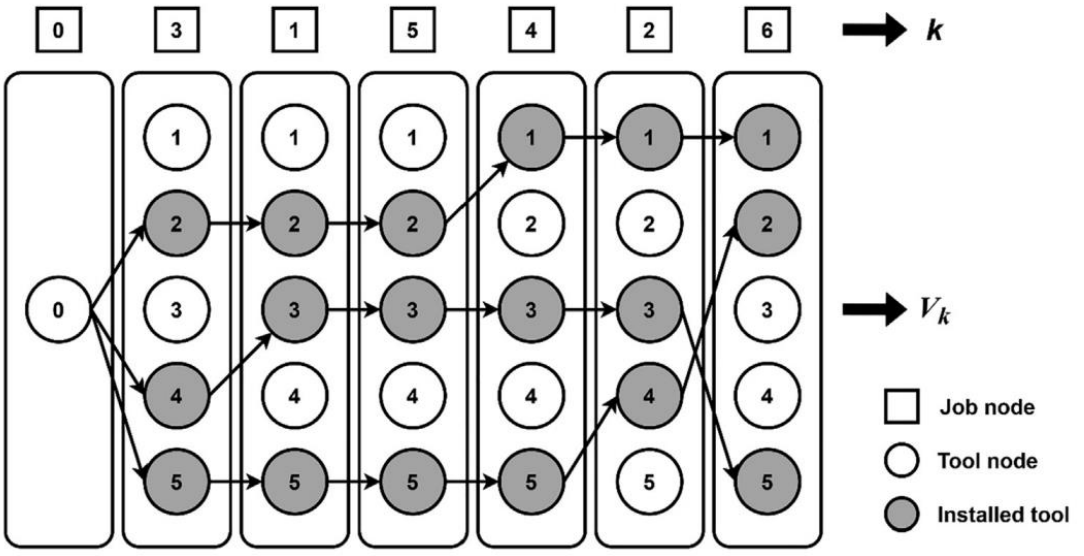


Figura 6 - Exemplo de um grafo de multicommodity flow (Mara et al., 2023)

Através deste modelo, é possível comparar a atribuição de ferramentas para cada vaga do carrossel em cada trabalho com um fluxo de uma só mercadoria. Para isso, é necessário apenas garantir que o número de fluxos que entra em cada nó corresponde à capacidade do carrossel

de ferramentas e que apenas uma ferramenta é armazenada em cada vaga. É possível, assim, criar um programa de mudança de ferramentas, como o que se encontra na Tabela 1 (Mara et al., 2023).

Tabela 1 - Escalonamento de trocas de ferramentas derivado da Figura 6

Trabalhos	**3**	**1**	**5**	**4**	**2**	**6**
Ferramentas	2	2	2	1	1	5
	4	3	3	3	3	2
	5	5	5	5	4	4

Portanto, a partir desta tabela é possível perceber quais as ferramentas inseridas na máquina em cada momento e, mais importante ainda, as trocas necessárias para passar de um trabalho para o seguinte, sinalizadas a laranja. Neste caso, considerou-se que o carrossel de ferramentas estava vazio no início, pelo que para o primeiro trabalho foi necessário colocar as 3 ferramentas (Mara et al., 2023).

Comparando as duas formulações abordadas em (Mara et al., 2023) ficou comprovado que o segundo método referido, o *multicommodity flow*, demonstra ter resultados com maior eficiência. Este permite obter a solução ótima para instâncias de problema de SSP dependentes da sequência de trabalhos em muito menos tempo computacional. Para além disso, o algoritmo semelhante ao caixeiro-viajante apenas consegue resolver otimamente instâncias com 10 ferramentas e uma capacidade de armazenamento na máquina de 5, enquanto este último algoritmo abordado consegue até 15 ferramentas e uma capacidade de 12 no carrossel.

2.4.2 Heurísticas

As heurísticas são das técnicas mais utilizadas para resolver problemas SSP, tal como já referido na subseção 2.2. O desempenho de várias abordagens depende da solução inicial utilizada. E, por vezes, as heurísticas servem também para isso, construindo primeiras soluções de qualidade. Em (Calmels, 2022) são desenvolvidas três heurísticas, duas baseadas em regras e uma com aleatoriedade, para construir a solução inicial de um problema SSP com máquinas paralelas não idênticas. As heurísticas construtivas são usadas para designar os trabalhos para cada máquina e para os sequenciar. Depois, para a mudança de ferramentas é usada a política KTNS, referida em (Tang & Denardo,1988). Esta política assenta em duas regras principais: só é inserida uma ferramenta se for necessária para o trabalho seguinte, e se o número de ferramentas preciso para o próximo trabalho for menor que a capacidade do carrossel de ferramentas da máquina, devem ser mantidas as ferramentas que são usadas nos trabalhos seguintes. Para se entender melhor esta última regra, na Tabela 2, encontra-se um exemplo de problema de escalonamento.

Tabela 2 - Exemplo de problema de escalonamento (Calmels, 2022)

Trabalhos	**1**	**2**	**3**	**4**	**5**	**6**
Ferramentas	1	1	2	1	3	1
	4	3	6	5	5	2
	8	5	7	7	8	4
	9		8	9		
Tempo Processamento M1	1	3	4	5	6	1
Tempo Processamento M2	-	4	-	-	3	1

Este exemplo conta com 6 trabalhos e duas máquinas, M1 e M2. Existem à disposição 9 ferramentas, mas os carrosséis têm apenas capacidade para 4 na M1 e 3 na M2. Convenciona-se o tempo de mudança de ferramentas de 1 unidade de tempo para a primeira máquina e 2 para a outra. Na tabela estão apresentadas as ferramentas necessárias para a execução de cada trabalho bem como o tempo que demora o seu processamento em cada máquina. Apenas os trabalhos que precisam de 3 ferramentas podem ser realizados na máquina 2 devido à capacidade do seu carrossel (Calmels, 2022). Na Figura 7, encontra-se então um possível escalonamento para este problema, neste caso, com o objetivo de minimizar o tempo de fluxo total.

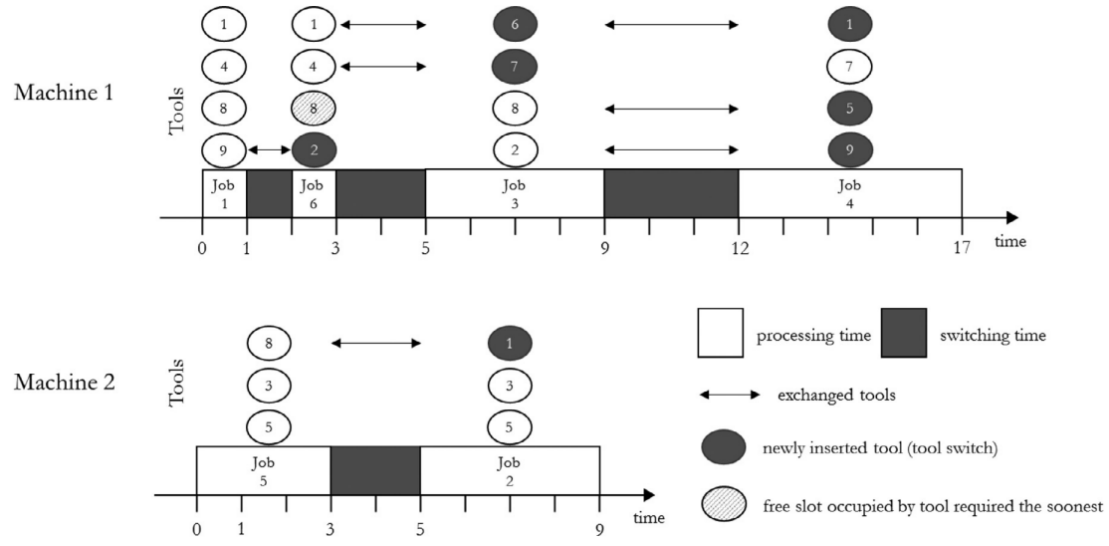


Figura 7 - Layout de possível solução para o problema da Tabela 2 (Calmels, 2022)

Assim, tal como é possível observar, apesar de o trabalho 6 não necessitar de 4 ferramentas, a ferramenta 8 permanece no espaço livre, uma vez que é necessário novamente no trabalho 3. Isto explica de forma visual e mais percetível o que se pretende atingir com a segunda regra da política KTNS (Calmels, 2022).

Começando pelas heurísticas baseadas em regras, o autor aborda duas: *Iterated Earliest Artificial Completion Time* (IEACT) e *Iterated Greatest Intersection* (IGI). De acordo com a primeira, a solução é construída do zero, acrescentando em cada iteração um trabalho no final da sequência de uma máquina. A escolha da máquina onde colocar o trabalho é feita tendo em conta o tempo de conclusão do último trabalho dessa, sendo selecionada a máquina com o tempo menor. O primeiro trabalho atribuído é o que tem um menor tempo de processamento. Após isto, os trabalhos não escalonados são avaliados e aquele com menor tempo de conclusão artificial passa a fazer parte da sequência parcial. Os empates são resolvidos de forma aleatória. Este cálculo do tempo de conclusão artificial é baseado no conceito de (Liu & Reeves, 2001). Segundo este, o tempo de conclusão artificial de um trabalho *p* é calculado tendo em conta que esse trabalho é escalonado logo depois de um dado trabalho *i* já colocado na sequência. A segunda heurística também constrói a solução do fim para o início, mas apresenta critérios diferentes para a escolha das máquinas e dos trabalhos. A máquina selecionada é aquela cujo último trabalho tem um tempo de conclusão artificial menor. Já para a escolha do primeiro trabalho, é necessário comparar o número de ferramentas necessárias para a realização de cada um deles e é selecionado o que apresenta o maior número, de forma a evitar mudanças posteriores. Depois, a regra é escolher o trabalho que possua o maior número de ferramentas em comum com o último trabalho. Mais uma vez, os empates são tratados aleatoriamente (Calmels, 2022).

A outra heurística abordada é a *Multi-Start-Random* (MSR), que gera trocas aleatórias entre todos os trabalhos. Posteriormente, com a sequência de trabalhos aloca-os às várias máquinas disponíveis. Começa-se pela primeira máquina e quando esta não puder receber um dado trabalho, atribuiu-se a uma outra máquina aleatória que cumpra as restrições de capacidade de armazenamento de ferramentas (Calmels, 2022).

Os resultados demonstraram que para a minimização do *makespan* e do número de trocas de ferramentas, a melhor heurística é a MSR para problemas pequenos. Se o objetivo for minimizar o tempo de fluxo total, o melhor método já e o IEACT, tanto para problemas pequenos como grandes. Para problemas grandes, o MSR registou os piores resultados, enquanto o IGI revelou minimizar melhor o número de mudanças de ferramentas. Relativamente ao tempo computacional necessário para chegar à solução, o MSR demonstrou ser significativamente lento comparando com outros métodos e, portanto, não consegue competir com esse nível de rapidez (Calmels, 2022). Isto revela que, apesar de os métodos exatos serem rotulados como mais lentos por analisarem exaustivamente a maioria das opções, as heurísticas também podem não corresponder com a rapidez desejada para a resolução destes problemas.

2.4.3 Meta-Heurísticas

Nos últimos anos, tem se procurado desenvolver algoritmos eficientes tanto na qualidade da solução como no tempo computacional necessário para a atingir. Apesar de se ter apostado mais em heurísticas, a utilização de meta-heurísticas para encontrar soluções tem despertado

um interesse crescente. A colónia de abelhas, o algoritmo genético e o simulated annealing são alguns exemplos de meta-heurísticas (Calmels, 2019).

Em (Dang et al., 2021), foi implementada uma integração de dois métodos: algoritmo genético e modelo de programação linear inteira. O primeiro é utilizado para atribuir os trabalhos às máquinas e sequenciá-los, enquanto o segundo resolve a componente de substituição de ferramentas. Este artigo também é um exemplo que comprova que alguns autores optam por dividir o problema SSP em partes e tratá-los separadamente. O objetivo principal desta combinação é utilizar as vantagens de ambos os métodos para encontrar soluções eficientes.

Em relação ao algoritmo genético, que é a meta-heurística utilizada, é importante referir como é feito o crossover, uma vez que este é o principal operador do algoritmo genético e tem um elevado efeito no seu desempenho. O crossover serve-se de dois cromossomas de cada vez para criar uma descendência com características de ambos os cromossomas pais. O método usado para escolher os cromossomas pais foi a seleção por torneios. São propostos dois crossovers: crossover combinado e crossover orientado para o problema. Contudo, cada um deles é composto por dois métodos. O primeiro junta o crossover de dois pontos e o crossover parcial mapeado adaptado, e é utilizado para criar os descendentes. O segundo é composto pela regra da data de entrega mais cedo (*Earliest Due Date* - EDD) e por uma heurística construtiva, e é usado depois de se encontrar uma nova solução ótima para orientar melhor a pesquisa (Dang et al., 2021).

Na Figura 8, é possível perceber melhor de que forma se insere a regra EDD no processo de crossover. Depois do crossover combinado, é aplicada à descendência a regra da data de entrega mais cedo nas partes que não sofreram alteração com o crossover. Por exemplo a subsequência 3-2 passa a ser 2-3 porque o trabalho 2 é para entregar a 13 e o trabalho 3 é só a 14.

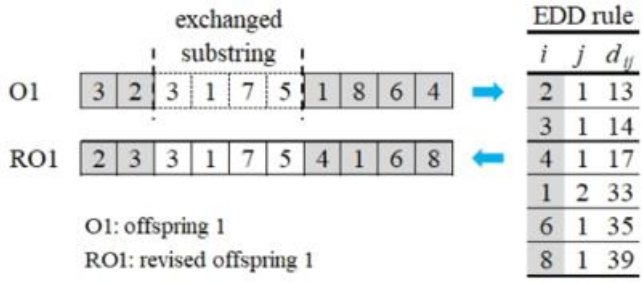


Figura 8 - Aplicação da regra EED no crossover (Dang et al., 2021)

Foram analisados alguns cenários para entender o impacto de certos parâmetros no resultado e também para comparar a meta-heurística apresentada com uma heurística abordada também no mesmo artigo, designada de Heurística do Praticante. É importante referir que o objetivo neste exemplo é minimizar tanto o atraso total como o tempo de setups (Dang et al., 2021).

Na Figura 9, encontra-se disposto o gráfico relativo a um dos cenários referidos no artigo, que avalia a influência da capacidade do carrossel de ferramentas nos resultados do SSP. A linha correspondente à meta-heurística referida é a laranja.

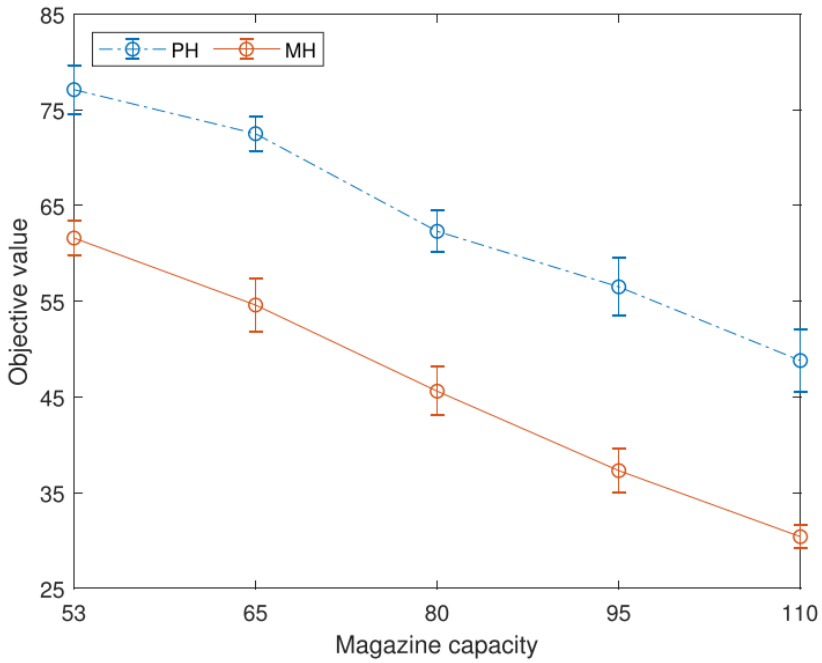


Figura 9 - Influência da capacidade do carrossel de ferramentas nos resultados do SSP (Dang et al., 2021)

A capacidade de armazenamento do carrossel de ferramentas das máquinas é um parâmetro com grande impacto e que afeta muito os tempos de troca, uma vez que se a capacidade fosse infinita e houvesse espaço para todas as ferramentas, este problema não existia, pois não seriam necessárias estas trocas. Assim, os resultados apresentados corroboram esta afirmação, já que à medida que a capacidade aumenta, o valor objetivo diminui. Este aumento de capacidade indica menos trocas, o que diminui o tempo de *setups* e pode ajudar a cumprir melhor os prazos (Dang et al., 2021).

Por fim, relativamente ao tempo computacional necessário para atingir a solução, a meta-heurística proposta é considerada aceitável dentro dos modelos atuais. É, também, robusta para os casos em que existem relações de precedências entre as operações produtivas (Dang et al., 2021). Através da observação do gráfico da Figura 9, é possível verificar que esta meta-heurística apresenta melhores resultados comparativamente com a heurística testada. Estas conclusões são um bom testemunho de que as meta-heurísticas têm características úteis para a resolução deste tipo de problemas.

Özpeynirci et al. (2016) abordaram dois modelos de programação linear inteira e uma meta-heurística. Desenvolveram o método *Tabu Search*, uma vez que com os modelos utilizados era impossível resolver instâncias de problema com mais de 15 trabalhos.

Na figura 10, está explicado o algoritmo do *Tabu Search* através de um fluxograma.

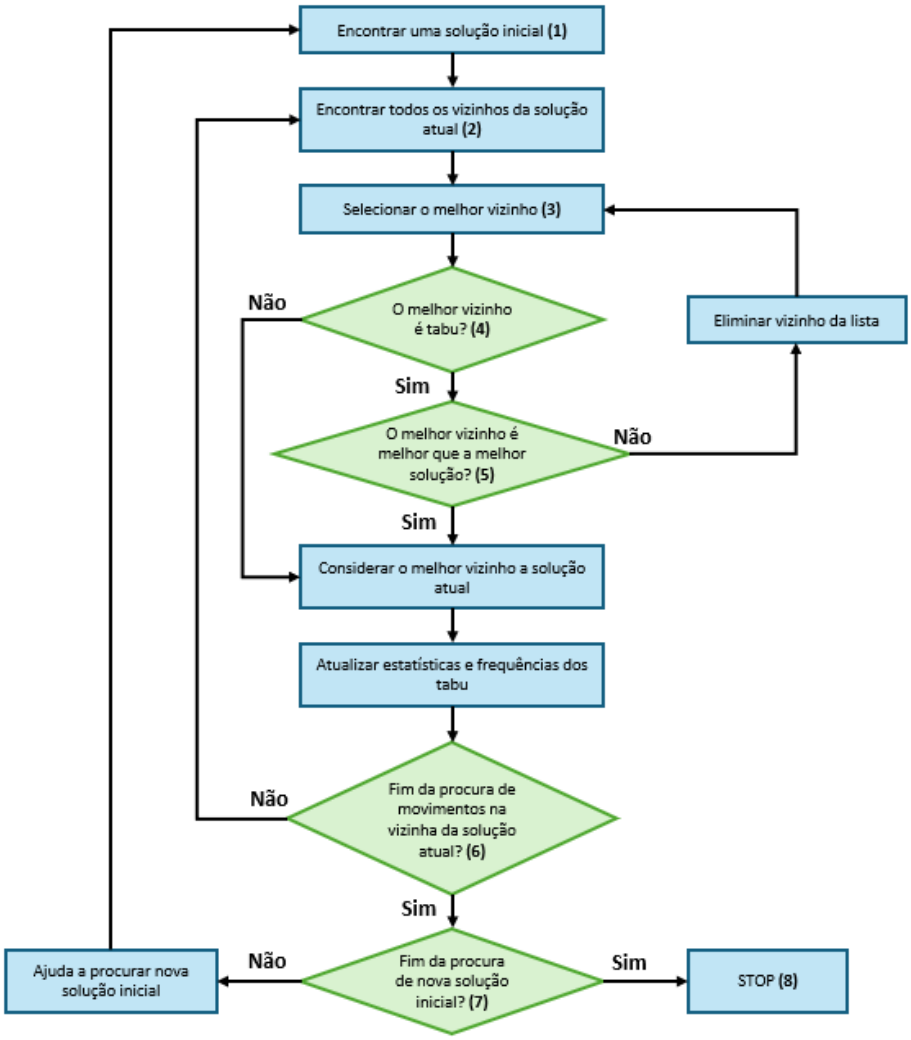


Figura 10 - Fluxograma do algoritmo Tabu Search (adaptado de (Özpeynirci et al., 2016))

O algoritmo começa com uma solução inicial válida (1) e, em cada iteração, avança-se para uma solução da vizinhança da solução atual (2). Normalmente, escolhe-se a melhor solução na vizinhança para o movimento seguinte (3). Não é permitido repetir movimentos durante um determinado número de iterações, conhecido como tempo tabu, e os movimentos proibidos são designados de tabu (4). O objetivo desta regra é evitar passar várias vezes pela mesma solução, ou seja, limitar a pesquisa a uma determinada área. Contudo, há uma exceção: pode ser feito um movimento tabu caso seja uma melhoria em relação à melhor solução encontrada até ao momento (5). O método *Tabu Search* utiliza duas estratégias: intensificação e diversificação. A primeira procura movimentos na vizinhança da solução atual. Após um determinado número de movimentos nessa vizinhança, o algoritmo salta para uma solução de outra vizinhança, através da estratégia da diversificação (6). Durante a fase de intensificação, a memória retém os movimentos realizados e ajuda a encontrar uma nova solução inicial numa outra parte do espaço de pesquisa (7). O algoritmo termina ao fim de um dado número de repetições da estratégia de diversificação (8) (Özpeynirci et al., 2016).

Esta meta-heurística obteve resultados quase ótimos, com desvios médios face à solução ótima entre 0% e 5,74%. Não se encontra um padrão significativo entre a qualidade das soluções e o tamanho do problema. Contudo, à medida que aumenta o número de trabalhos, o tempo para a solução também aumenta. Mesmo assim, é previsível que este algoritmo permita encontrar soluções de qualidade e em tempos razoáveis, uma vez que a qualidade das soluções é robusta (Özpeynirci et al., 2016).

2.4.4 Multiobjetivo

A preferência por abordagens multiobjectivo para a resolução de problemas do tipo SSP não tem sido muito grande ao longo dos anos, apesar de existirem alguns casos pontuais. Pode considerar-se esta vertente de resolução ainda pouco aprofundada, comparativamente às funções objetivo com apenas um objetivo. Contudo, acrescenta um pouco mais de realidade ao caso, uma vez que no mundo real, normalmente, são vários os objetivos a atingir, mesmo que com pesos diferentes.

O indicador mais utilizado neste tipo de problemas de escalonamento é o *makespan*, mas existem muitas outras possibilidades já referidas ao longo da análise de literatura feita, como por exemplo, os tempos de troca de ferramentas e os atrasos nas entregas. Num ambiente industrial, é comum os produtos fabricados estarem sujeitos a uma data-limite de produção, com vista a corresponder com as necessidades dos clientes. Por isso, o desempenho do fabricante quanto ao cumprimento das datas de entrega é um critério relevante para a avaliação do desempenho de um sistema produtivo. Por essa mesma razão, surgiu o interesse de avaliar também o impacto deste objetivo nos problemas SSP. No artigo de (Rifai et al., 2022-b) é, então, desenvolvido um modelo de otimização multiobjectivo que pretende minimizar simultaneamente o tempo de trocas de ferramentas nas máquinas e o atraso do processo.

Nos problemas gerais de escalonamento de trabalhos, normalmente um elevado tempo de *setup* implica uma maior possibilidade de atrasos, o que faz com que estas duas variáveis tenham uma relação linear positiva, já que o aumento de uma pode levar ao aumento da outra. Porém, nos problemas SSP pode verificar-se o oposto. Isso deve-se à componente da troca de ferramentas. Para evitar sofrer atrasos, habitualmente colocam-se os trabalhos com datas mais próximas primeiro, priorizando-se o critério da data-limite. Mas essa passagem dos trabalhos para uma posição anterior na sequência pode implicar o aumento dos tempos de mudança de ferramentas. Isto pode ser ainda mais acentuado nos problemas SSP que têm tempos de trocas dependentes da sequência, uma vez que esta alteração de trabalhos pode implicar tempos de mudança de ferramentas superiores aos que seriam escolhidos caso o objetivo principal fosse minimizar estes tempos (Rifai et al., 2022-b).

Solimanpur & Rastgordani (2012) também aplicaram um método de otimização das colónias de formigas com função multiobjectivo de minimizar tanto as trocas de ferramentas como a taxa de rotação do carrossel. Mais uma vez, estes fatores estão relacionados na medida em que quantas mais trocas de ferramentas forem feitas, maior é a rotação do carrossel.

2.5 Modelo de referência

O modelo que serve de referência para o trabalho desenvolvido é o M1 proposto por Beezão et al. (2017). Neste subcapítulo, são apenas apresentados os parâmetros, as variáveis de decisão e o modelo. Posteriormente, será explicado mais ao detalhe o modelo e as suas restrições, incluindo as alterações feitas no novo modelo desenvolvido.

Os parâmetros utilizados são:

- Um conjunto de trabalhos $J = \{1, \dots, n\}$;

- Um conjunto de máquinas paralelas idênticas $M = \{1, \dots, p\}$;

- p_j corresponde ao tempo de processamento do trabalho $j \in J$;

- Um conjunto de ferramentas disponíveis $T = \{1, \dots, l\}$;

- C é a capacidade máxima do carrossel de ferramentas de cada máquina (constante);

- $\bar{t}$ é o tempo necessário para efetuar uma troca de ferramenta, seja uma inserção ou uma remoção (constante);

- J_t é o conjunto de trabalhos j que utilizam a ferramenta $t \in T$.

As variáveis de decisão são as seguintes:

- $W_{jkm} = 1$ se o trabalho j for executado na posição k da sequência da máquina m, $W_{jkm} = 0$ caso contrário;

- $Y_{tkm} = 1$ se a ferramenta t for utilizada na execução do trabalho que está na posição k da sequência da máquina m, $Y_{tkm} = 0$ caso contrário;

- $Z_{tkm} = 1$ se a ferramenta t for inserida na máquina m exatamente antes da execução do trabalho na posição k da sequência, $Z_{tkm} = 0$ caso contrário;

- Δ_m representa o tempo de conclusão da máquina $m \in M$;

- Δ é o *makespan*, que corresponde ao maior valor dos Δ_m.

A função objetivo (1) e as restrições (2) – (9) utilizadas no modelo são apresentadas de seguida:

$$FO: Min\ \Delta \tag{1}$$

$$\sum_{j=1}^{n} W_{jkm} \leq 1 \qquad k \in J, m \in M \tag{2}$$

$$\sum_{m=1}^{p} \sum_{k=1}^{n} W_{jkm} = 1 \qquad j \in J \tag{3}$$

$$\sum_{j=1}^{n} W_{jkm} \leq \sum_{j=1}^{n} W_{j(k-1)m} \qquad k \in J\backslash\{1\}, m \in M \tag{4}$$

$$\sum_{j \in J_t} W_{jkm} \leq Y_{tkm} \qquad t \in T, k \in J, m \in M \tag{5}$$

$$\sum_{t=1}^{l} Y_{tkm} \leq C \qquad k \in J, m \in M \tag{6}$$

$$Y_{tkm} - Y_{t(k-1)m} \leq Z_{tkm} \qquad t \in T, k \in J\backslash\{1\}, m \in M \tag{7}$$

$$\sum_{k=1}^{n} \sum_{j=1}^{n} p_j W_{jkm} + \bar{t} \sum_{k=2}^{n} \sum_{t=1}^{l} Z_{tkm} \leq \Delta_m \qquad m \in M \tag{8}$$

$$\Delta_m \leq \Delta \qquad m \in M \tag{9}$$

$$W_{jkm} \in \{0,1\} \qquad j \in J, k \in J, m \in M \tag{10}$$

$$Y_{tkm} \in \{0,1\} \qquad t \in T, k \in J, m \in M \tag{11}$$

$$Z_{tkm} \in \{0,1\} \qquad t \in T, k \in J, m \in M \tag{12}$$

$$\Delta_m \geq 0, \Delta \geq 0 \qquad m \in M \tag{13}$$

3 Modelo de Programação Linear Inteira Mista

Este capítulo encontra-se dividido em 4 subcapítulos. No primeiro, é descrito o problema detalhadamente. No subcapítulo seguinte, são referidas as alterações principais e as adições feitas no novo modelo desenvolvido face ao modelo de referência e em Modelo Matemático para o SSP é explicado ao pormenor o novo modelo criado, desde as variáveis utilizadas às restrições. São ainda expostas as limitações do modelo, que motivam o capítulo seguinte.

3.1 Descrição do problema

Este estudo tem como foco uma empresa da indústria metalúrgica com mais de 100 anos de experiência na área. Comercializa vários tipos de produtos, como soluções de automação, sistemas de maquinação e ferramentas de corte de alta precisão. Esta última vertente é a mais importante para o caso, uma vez que o problema em análise se refere à produção das ferramentas de corte de alta precisão. Apesar deste estudo se focar mais neste cenário em específico, as abordagens apresentadas são possíveis de aplicar noutras indústrias com características semelhantes.

Esta empresa tem uma presença global e enquadra-se num mercado muito competitivo, onde se torna fundamental ter um escalonamento que vá de encontro às necessidades dos seus clientes. Por questões de confidencialidade com a empresa, não é possível divulgar a identidade da mesma.

A empresa funciona 5 dias por semana em 3 turnos de 8 horas cada com 4 operadores. Existem *m* máquinas paralelas idênticas, capazes de realizar qualquer trabalho, e do tipo CNC, ou seja, que executam várias tarefas, como torneamento e fresagem. Cada trabalho deve ser executado somente numa máquina e não são permitidas interrupções. Não existem quaisquer relações de precedência entre trabalhos. Como as máquinas são idênticas, os tempos de processamento dos trabalhos e as ferramentas necessárias para a sua execução são iguais para todas as

máquinas. As máquinas têm um limite de tempo de funcionamento semanal de 100 horas, podendo este apenas ser ultrapassado no caso de um trabalho só ser mais longo que este tempo. Os carrosséis de ferramentas das máquinas têm um limite de capacidade, também igual entre si.

Para a realização de cada trabalho, são utilizadas várias ferramentas, até o máximo da capacidade do carrossel, ou seja, existem trabalhos que necessitam de apenas duas ferramentas e outros que já precisam de seis. Cada trabalho diz respeito à produção de uma dada quantidade de peças, não sempre unitária. Por vezes, devido à limitação de capacidade do carrossel, para a realização de um dado trabalho, é necessário remover algumas ferramentas do carrossel da máquina que tinham sido colocadas para o trabalho anterior, para ser possível colocar as ferramentas necessárias para este novo trabalho. O tempo requerido para inserir ou remover alguma ferramenta é igual para todas as máquinas e para todas as ferramentas, mas este tempo entre trabalhos é dependente da quantidade de trocas que é necessário fazer. Por isso mesmo, considera-se os tempos de *setup* dependentes da sequência.

No início de cada semana de trabalho, considera-se que não existe nenhuma ferramenta inserida nos carrosséis das máquinas e que as hastes destas não estão calibradas para nenhum diâmetro em específico. Portanto, antes de começar a produção semanal é preciso fazer as etapas de preparação das máquinas, de inserção de ferramentas e definição do diâmetro da haste.

Durante a execução dos trabalhos, não é necessário auxílio de nenhum operador, apenas nas passagens entre trabalhos. É necessário um operador por máquina sempre que é preciso efetuar alguma alteração na máquina. Estas alterações incluem as trocas de ferramentas, a alteração do diâmetro da haste e o *setup* entre trabalhos para preparar o programa a executar. O diâmetro da peça a produzir tem impacto na máquina, uma vez que é necessário ajustar também o diâmetro da sua haste. O tempo requerido para este ajuste é sempre igual, mas só acontece quando o diâmetro das peças dos trabalhos adjacentes é diferente. Já o tempo de *setup* entre trabalhos é diferente consoante o trabalho que se inicia posteriormente. Este *setup* está relacionado com a categoria de produtos, se forem da mesma categoria, não existe este tempo. Quando os dois trabalhos são de categorias diferentes, é realizado o *setup*, cujo tempo é dependente da sequência. O cálculo deste tempo de *setup* é resultado do produto entre o tempo de produção unitário da ordem que se segue e uma constante K, definida pela empresa como K=2. A categoria dos materiais está definida no código do material, caso a parte inicial do código seja igual, significa que esses dois produtos são da mesma categoria.

O objetivo deste departamento de produção de ferramentas de corte de alta precisão é definir um plano de produção, minimizando o *makespan*, ou seja, a data de conclusão de todos os trabalhos e, ainda, reduzindo o número de trocas de ferramentas.

3.2 Alterações e adições no novo modelo

Neste subtópico são apresentadas as alterações e adições mais relevantes feitas decorrente da criação do novo modelo, face ao modelo usado como referência. Portanto, as modificações foram as seguintes: uma melhoria na restrição da necessidade de ferramentas nas máquinas em cada posição da sequência, a restruturação das restrições de inserção de ferramentas, a inclusão da parte de remoção de ferramentas dos carrosséis das máquinas, a introdução da modelação de *setups* sequenciais e a incrementação de configurações iniciais das máquinas.

3.2.1 Melhoria na restrição da necessidade de ferramentas nas máquinas

A restrição (5) do modelo de referência impõe que todas as ferramentas necessárias para um trabalho estão disponíveis aquando da sua realização. No entanto, a maneira como está escrita permite que a variável que indica se é necessária ou não a ferramenta (Y_{tkm}) informe que sim, mesmo não existindo nenhum trabalho que utilize aquela ferramenta naquela posição da sequência de trabalhos daquela máquina. Isto é visível através da Tabela 3, que mostra a tabela de verdade decorrente da restrição (5) do modelo.

Tabela 3 - Tabela de Verdade da restrição (5) do modelo de referência

$\sum_{j \in J_t} W_{jkm}$	$\leq$	Y_{tkm}
0	$\leq$	0 ou 1
1	$\leq$	1

A solução encontrada para este problema foi substituir o operador lógico da restrição, passando de $\leq$ para $=$, conforme se encontra na restrição (18) do modelo desenvolvido neste trabalho. Desta forma, sempre que existir um trabalho que necessite da ferramenta a variável apresenta o valor binário 1. Caso contrário, apresenta o valor 0.

3.2.2 Restruturação das restrições de inserção de ferramentas

Em relação à colocação de ferramentas no carrossel das máquinas, no modelo de referência apenas existe a restrição (7) que indica se a ferramenta foi colocada no carrossel exatamente antes do trabalho atual. Porém esta restrição não é suficiente, tal como se pode ver pela Tabela 4, que apresenta a tabela de verdade dessa restrição. Através dela, é possível perceber que existem várias situações em que a informação da inserção pode estar incorreta.

Tabela 4 - Tabela de verdade da restrição (7) do modelo de referência

Y_{tkm}	−	$Y_{t(k-1)m}$	≤	Z_{tkm}
1	−	0	≤	1
0	−	1	≤	0 ou 1
1	−	1	≤	0 ou 1
0	−	0	≤	0 ou 1

Por essa mesma razão, foi adicionada a restrição (21) ao modelo desenvolvido que garante que a variável Z_{tkm} não é igual a 1 quando a variável Y_{tkm} é igual a 0, ou seja, que não é colocada uma dada ferramenta quando ela não é necessária. Foi adicionada, ainda, a restrição (22) para que não seja colocada uma ferramenta caso não seja necessária para aquele trabalho.

3.2.3 Etapa de remoção de ferramentas do carrossel da máquina

No modelo apresentado como referência, existem apenas variáveis de decisão e restrições para a etapa de inserção de ferramentas no carrossel da máquina, não existindo nenhuma contagem do número de vezes em que é necessário retirar uma ferramenta para ser possível inserir outra, nem do tempo requerido para isso. Contudo, isso também afeta o tempo de conclusão dos trabalhos e, por isso, é essencial tê-lo em conta.

Posto isto, foram criadas as restrições (24) – (26) do mesmo género que as restrições de inserção de ferramentas, mas com a lógica contrária. Assim, o que se pretende é que só se retire uma dada ferramenta caso ela fosse necessária no trabalho anterior e já não seja no trabalho atual. Isso é o que se pretende provar pela tabela de verdade da Tabela 5.

Tabela 5 - Tabela de verdade pretendida para as restrições de remoção de ferramentas

$Y_{t(k-1)m}$	Y_{tkm}	=>	X_{tkm}
1	0	=>	1
0	1	=>	0
1	1	=>	0
0	0	=>	0

Para o desenvolvimento destas restrições foi necessário criar uma variável de decisão, que apresente o valor 1 quando for para retirar a ferramenta e 0 caso contrário. Assim, foi criada a variável X_{tkm}, que é semelhante à de inserção de ferramentas.

3.2.4 Introdução da modelação de setups sequenciais

Todas as etapas de *setup* neste problema são dependentes da sequência. Neste caso em concreto, o *setup* engloba três tarefas principais: a troca de ferramentas, a alteração do

diâmetro da haste da máquina e outros ajustes decorrentes da categoria do produto. Em relação à troca de ferramentas, esse tópico já foi discutido nos subcapítulos anteriores.

Cada trabalho engloba um produto, que apresenta um determinado diâmetro e pertence a uma dada categoria. Por essa razão, ao passar de um diâmetro para outro diferente, é preciso modificar a haste da máquina para corresponder ao diâmetro certo. A restrição do novo modelo (27) garante exatamente isso, ou seja, caso o diâmetro seja o mesmo, não é adicionado nenhum tempo, caso seja diferente, é adicionado o tempo necessário para fazer as alterações. Esta quantidade de tempo é fixa.

O mesmo acontece para a categoria do produto. Isto é, quando a categoria é a mesma, não é necessário nenhum ajuste, logo não é adicionado nenhum tempo. Já quando a categoria é diferente, são realizadas algumas alterações na máquina e por isso, é adicionada essa componente de *setup*. Este tempo, ao contrário do que se passa com o diâmetro, não é fixo. Está diretamente relacionado com o trabalho que se inicia no momento seguinte, mais precisamente com o tempo de produção unitário, ou seja, com o tempo necessário para produzir uma unidade daquele material. Contudo, está sujeito a uma constante de proporcionalidade direta, que é designada por K. A restrição adicionada ao modelo que reflete isto é a equação (29).

Estas novas restrições implicaram a criação de novas variáveis de decisão. Como ambas as situações estão diretamente relacionadas com os trabalhos, convencionou-se utilizar uma configuração semelhante à variável dos trabalhos, ou seja, criaram-se as seguintes variáveis: H_{jkm} para o diâmetro e C_{jkm} para a categoria.

3.2.5 Configurações iniciais das máquinas

Tal como já foi referido no subcapítulo 3.1, no início de cada semana de trabalho, é necessário preparar as máquinas para começar o trabalho. Considera-se que não existe nenhuma ferramenta inserida nos carrosséis das máquinas e que as hastes destas não estão calibradas para nenhum diâmetro em específico. Portanto, mediante o primeiro trabalho a realizar em cada máquina, tem de ser realizado o ajuste do diâmetro da haste e têm de ser colocadas as ferramentas necessárias para esse trabalho.

Desta forma, foram criadas as restrições (23) e (28) que são exclusivamente para a primeira posição da sequência de cada máquina. Através delas, são garantidas as configurações iniciais das máquinas.

3.3 O Modelo Matemático para o SSP

Neste subcapítulo é apresentado o modelo desenvolvido para resolver o problema de escalonamento de trabalhos com a troca de ferramentas nas máquinas. É proposto um modelo de programação linear inteira mista (PLIM), com o objetivo de minimizar o *makespan*, ou seja,

o tempo de conclusão dos trabalhos. Em primeiro lugar, são definidas as variáveis de decisão e os parâmetros utilizados. Posteriormente, é explicado ao detalhe a formulação matemática, incluindo as suas restrições.

3.3.1 Definição das variáveis de decisão e dos parâmetros

Os parâmetros e conjuntos utilizados são os seguintes:

- Um conjunto de j trabalhos, denotado como $J = \{1, \dots, n\}$;
- Um conjunto de m máquinas paralelas idênticas, cujo conjunto é representado por $M = \{1, \dots, p\}$;
- p_j corresponde ao tempo de processamento do trabalho $j \in J$ (em horas);
- Um conjunto de t ferramentas disponíveis, com $T = \{1, \dots, l\}$;
- C é a capacidade máxima do carrossel de ferramentas de cada máquina;
- $\bar{t}$ é o tempo necessário (em horas) para efetuar uma troca de ferramenta, seja uma inserção ou uma remoção;
- J_t é o conjunto de trabalhos j que utilizam a ferramenta $t \in T$;
- D_j é o diâmetro do material a produzir no trabalho j (em milímetros);
- U_j é o tempo de produção unitário do material a produzir no trabalho j (em horas);
- K é uma constante de proporcionalidade usada para calcular o tempo de setup entre trabalhos, decorrente da categoria do material em produção;
- E_j corresponde à categoria do material fabricado no trabalho j;
- $\bar{s}$ é o tempo necessário (em horas) para efetuar uma alteração de diâmetro da haste da máquina.

Os tempos de troca de ferramentas e de alteração do diâmetro são fixos, é um pressuposto da empresa cujo problema está em estudo. Além disso, foi convencionado trabalhar na unidade temporal hora, portanto todos os tempos apresentam-se em horas.

As variáveis de decisão utilizadas podem dividir-se em dois tipos principais: binárias e contínuas. Na Tabela 6, apresentam-se as variáveis utilizadas.

Tabela 6 - Variáveis de decisão do modelo

Variáveis de decisão	Descrição
W_{jkm}	$\begin{cases} 1, \text{se o trabalho j for executado na posição k da máquina m} \\ 0, \text{caso contrário} \end{cases}$
Y_{tkm}	$\begin{cases} 1, \text{se a ferramenta t estiver presente na máquina m durante a realização do trabalho na posição k} \\ 0, \text{caso contrário} \end{cases}$
Z_{tkm}	$\begin{cases} 1, \text{se a ferramenta t for inserida na máquina m exatamente antes do trabalho na posição k} \\ 0, \text{caso contrário} \end{cases}$
X_{tkm}	$\begin{cases} 1, \text{se a ferramenta t for removida da máquina m antes da realização do trabalho na posição k} \\ 0, \text{caso contrário} \end{cases}$
H_{jkm}	$\begin{cases} 1, \text{se houver alteração do diâmetro da haste da máquina m antes da realização do trabalho j na posição k} \\ 0, \text{caso contrário} \end{cases}$
C_{jkm}	$\begin{cases} 1, \text{se houver mudança de categoria de trabalho antes da realização do trabalho j na posição k da máquina m} \\ 0, \text{caso contrário} \end{cases}$
Δ_m	Tempo de conclusão da máquina m $(m \in M)$
Δ	*makespan*

3.3.2 Formulação matemática

Para este modelo PLIM é utilizada a formulação matemática, conforme abaixo.

$$FO: Min\ Z = \Delta \tag{14}$$

$$\sum_{j=1}^{n} W_{jkm} \leq 1 \qquad k \in J, m \in M \tag{15}$$

$$\sum_{m=1}^{p} \sum_{k=1}^{n} W_{jkm} = 1 \qquad j \in J \tag{16}$$

$$\sum_{j=1}^{n} W_{jkm} \leq \sum_{j=1}^{n} W_{j(k-1)m} \qquad k \in J \backslash \{1\}, m \in M \tag{17}$$

$$\sum_{j \in J_t} W_{jkm} = Y_{tkm} \qquad t \in T, k \in J, m \in M \tag{18}$$

$$\sum_{t=1}^{l} Y_{tkm} \leq C \qquad k \in J, m \in M \tag{19}$$

$$Y_{tkm} - Y_{t(k-1)m} \leq Z_{tkm} \qquad t \in T, k \in J \backslash \{1\}, m \in M \tag{20}$$

$$Z_{tkm} \leq Y_{tkm} \qquad t \in T, k \in J, m \in M \tag{21}$$

$$Z_{tkm} \leq 1 - Y_{t(k-1)m} \qquad t \in T, k \in J\backslash\{1\}, m \in M \tag{22}$$

$$Y_{t1m} = Z_{t1m} \qquad t \in T, m \in M \tag{23}$$

$$Y_{t(k-1)m} - Y_{tkm} \leq X_{tkm} \qquad t \in T, k \in J\backslash\{1\}, m \in M \tag{24}$$

$$X_{tkm} \leq Y_{t(k-1)m} \qquad t \in T, k \in J\backslash\{1\}, m \in M \tag{25}$$

$$X_{tkm} \leq 1 - Y_{tkm} \qquad t \in T, k \in J, m \in M \tag{26}$$

$$\begin{cases} W_{jkm} + W_{j'(k-1)m} - 1 \leq H_{jkm} & se\ D_j \neq D_{j'} \\ H_{jkm} \leq W_{jkm} \end{cases} \qquad j, j' \in J, k \in J\backslash\{1\}, m \in M \tag{27}$$

$$W_{j1m} = H_{j1m} \qquad j \in J, m \in M \tag{28}$$

$$\begin{cases} W_{jkm} + W_{j'(k-1)m} - 1 \leq C_{jkm} & se\ E_j \neq E_{j'} \\ C_{jkm} \leq W_{jkm} \end{cases} \qquad j, j' \in J, k \in J\backslash\{1\}, m \in M \tag{29}$$

$$\sum_{k=1}^{n}\sum_{j=1}^{n} p_j W_{jkm} + \bar{t}\left(\sum_{k=1}^{n}\sum_{t=1}^{l} Z_{tkm} + \sum_{k=2}^{n}\sum_{t=1}^{l} X_{tkm}\right) + \bar{s}\sum_{k=1}^{n}\sum_{j=1}^{n} H_{jkm} + \sum_{k=2}^{n}\sum_{j=1}^{n} KU_j C_{jkm} \leq \Delta_m \tag{30}$$

$$\Delta_m \leq \Delta \qquad m \in M \tag{31}$$

$$W_{jkm} \in \{0,1\} \qquad j \in J, k \in J, m \in M \tag{32}$$

$$Y_{tkm} \in \{0,1\} \qquad t \in T, k \in J, m \in M \tag{33}$$

$$Z_{tkm} \in \{0,1\} \qquad t \in T, k \in J, m \in M \tag{34}$$

$$X_{tkm} \in \{0,1\} \qquad t \in T, k \in J, m \in M \tag{35}$$

$$H_{jkm} \in \{0,1\} \qquad j \in J, k \in J, m \in M \tag{36}$$

$$C_{jkm} \in \{0,1\} \qquad j \in J, k \in J, m \in M \tag{37}$$

$$\Delta_m \geq 0, \Delta \geq 0 \qquad m \in M \tag{38}$$

A função objetivo do modelo (14) minimiza o *makespan*. As restrições (15) e (16) estão relacionadas com a atribuição dos trabalhos às máquinas, enquanto a restrição (17) assegura que um trabalho só pode ser executado pela máquina m na posição k depois do trabalho $(k-1)$ dessa máquina ter terminado.

As restrições (18) e (19) estão ambas relacionadas com as ferramentas: a primeira impõe que todas as ferramentas necessárias para um trabalho estão disponíveis aquando da sua realização e a segunda limita a capacidade do carrossel de ferramentas de cada máquina. A inserção das ferramentas na máquina é gerida pelas restrições (20), (21), (22) e (23). A restrição (23) é mais específica, dizendo respeito apenas ao que é necessário colocar na máquina antes do primeiro trabalho. As restrições (24), (25) e (26) são semelhantes às anteriores, mas abordam a etapa de remoção das ferramentas.

As restrições (27) e (28) estão ambas relacionada com a troca de diâmetro e controlam os tempos utilizados para as alterações necessárias na máquina. Contudo, a restrição (28) é mais

específica, porque reflete uma configuração inicial da máquina, ou seja, antes do primeiro trabalho. Já a restrição (29) reflete os tempos de *setup* entre trabalhos sempre que a categoria do material a produzir altera. A restrição (30) calcula o tempo de conclusão de cada máquina e o *makespan* é definido pela restrição (31). As restrições (32) – (37) denotam as restrições de integralidade e as restrições da equação (38) indicam a não negatividade das variáveis de tempo de conclusão e *makespan*.

3.4 Limitações do Modelo Matemático

O modelo PLIM apresentado não é indexado ao tempo, utiliza a apenas a posição do trabalho na sequência de tarefas em cada máquina. Por essa razão, não foi possível implementar a restrição do número de operadores disponíveis para fazer os *setups* das máquinas nem do tempo limite de funcionamento das máquinas. Este número iria variar ao longo do tempo e não da posição do trabalho na máquina.

4 Meta-Heurística e Heurística Construtiva

Este capítulo encontra-se dividido em dois temas principais: a meta-heurística e a heurística construtiva. Assim, o primeiro subcapítulo explica a meta-heurística, Simulated Annealing, desenvolvida para combater as dificuldades encontradas com o modelo, referidas no subcapítulo 3.4. O subcapítulo seguinte apresenta a parametrização feita para o Simulated Annealing aplicado. Por fim, é explicada a heurística construtiva criada, bem como a regra por trás dela.

4.1 Simulated Annealing

Quando se trata de problemas que exigem esforços computacionais grandes, uma das abordagens mais utilizadas são as meta-heurísticas. De uma forma genérica, estas podem ser definidas como regras gerais que, independentemente do problema, derivam algoritmos de otimização (Franzin & Stützle, 2019).

Assim, por forma a tentar dar resposta às condições do problema que o modelo matemático não conseguiu resolver, foi desenvolvida neste projeto uma meta-heurística, mais precisamente o Simulated Annealing (SA).

O Simulated Annealing é uma das meta-heurísticas mais antigas e ao longo dos anos já foi adaptado por vários autores para resolver problemas de otimização combinatória. De facto, aquando da realização do artigo de (Franzin & Stützle, 2019), já existiam mais de 6 000 artigos com a palavra-chave “Simulated Annealing” no título e, se a pesquisa for alargada ao resumo, aumenta exponencialmente para 30 000 artigos.

O SA é um algoritmo de pesquisa local estocástico que parte de uma solução inicial e iteração a iteração vai explorando a sua vizinhança. As soluções melhores são sempre aceites e as piores dependem de uma probabilidade que é calculada com base num parâmetro, designado

temperatura. Esta característica do método é o que o distingue dos restantes, uma vez que lhe permite investigar vizinhanças que à partida já seriam descartadas. A criação da meta-heurística como é conhecida aos dias de hoje para a resolução de problemas de otimização combinatória foi feita entre 1983 e 1985. Contudo, a inspiração por detrás deste método remonta a 1953, quando Metropolis et al. propuseram utilizar Monte Carlo para resolver problemas de estado de sistemas físicos constituídos por partículas. Comparando a ideia inicial com o que é utilizado atualmente, a temperatura física seria o parâmetro da temperatura, o estado físico do sistema seria a solução candidata, o estado fundamental seria a melhor solução encontrada até ao momento e uma mudança de estado significaria uma procura de novas soluções na vizinhança (Franzin & Stützle, 2019).

Esta meta-heurística pode ser composta em várias fases principais:

- O cálculo do valor(es) objetivo(s) de uma dada solução;
- O critério de aceitação de novas soluções;
- O critério de exploração de outras soluções na vizinhança;
- A definição de critérios de paragem e de temperatura;
- O próprio algoritmo simulated annealing.

Neste trabalho em concreto, desenvolveram-se duas versões do método: a primeira, mais simplificada, com características similares ao modelo matemático para ser possível comparar resultados; a segunda, com as restrições adicionais que não estavam incluídas no modelo. A primeira versão, contém apenas um cenário de objetivo, a minimização do *makespan*. Já a segunda versão, contém vários cenários de objetivo, desde *makespan*, troca de ferramentas e, até, função multiobjectivo. Por essa razão, a meta-heurística criada vai ser explicada para a primeira versão, sendo referidas as alterações realizadas para a segunda versão.

Começando pelo próprio algoritmo em si, na Figura 11 é apresentado o pseudocódigo do Simulated Annealing.

```
procedure simulated_annealing
        #First Solution
        read first solution (randomly generated)
        current_solution ← first solution
        best_solution ← current_solution
        current_cost ← objective_function (current_solution)
        best_cost ← current_cost
        temperature ← initial_temperature
        #Iterations
        for i = 1, …, iterations do
                while restriction is not ok do
                        new_solution ← neighborhood (current_solution)
                        evaluate the validity of the solution
                end;
                next_cost ← objective_function (new_solution)
                probability ← acceptance_probability (current_cost, next_cost, temperature)
                if probability > random value then:
                        current_solution ← new_solution
                        current_cost ← next_cost
                if current_cost < best_cost then:
                        best_solution ← current_solution
                        best_cost ← current_cost
                variation = abs(current_cost - best_cost)/best_cost
                if varaiation ≤ variation_margin then:
                        iterations_without_change = iterations_without_change + 1
                else
                        iterations_without_change = 0
                if iterations_without_change ≥ max_iterations_without_change then:
                        Stop the process
                if the number i is multiple of L then:
                        temperature = temperature x α
        end;
        return best_solution, best_cost
end simulated_annealing
```

Figura 11 - Pseudocódigo do Simulated Annealing

Portanto, pela observação da Figura 11, percebe-se que o programa inicia com uma solução inicial gerada de forma aleatória. Depois disso, inicia-se um ciclo com um determinado número de iterações, onde são geradas novas soluções através da função de vizinhança. Estas novas soluções ainda têm de ser validadas antes do passo seguinte. A validação é para cumprir o requisito do tempo limite de funcionamento das máquinas. Posto isto, é calculado o valor ou valores objetivo da nova solução. Este valor é posteriormente usado na função

acceptance_probability que estipula a probabilidade de aceitação daquela solução. Mediante a comparação entre essa probabilidade e um valor aleatório, a solução passa a ser a corrente. Caso seja melhor que a melhor solução encontrada até ao momento, passa a ser essa a melhor solução. Neste passo, caso seja função multiobjectivo, é um pouco mais complexo, uma vez que se tem de comparar duas ou mais métricas diferentes. As funções multiobjectivo também podem ser diferentes, isto é, cada objetivo pode ter a mesma importância ou importâncias diferentes, fazendo com que uma das métricas tenha mais relevância que as restantes. Por exemplo, caso o primeiro objetivo seja minimizar o *makespan* e o segundo seja minimizar o número de trocas de ferramentas, a comparação tem dois níveis: caso o *makespan* seja melhor, é sempre passado a solução melhor; fora esse caso, só é passado a solução melhor se reduzir o número de trocas, mas o *makespan* se mantiver igual. A temperatura é reduzida com base no fator de arrefecimento α (predefinido nos parâmetros) e a cada L iterações (parâmetro L também definido desde início). É importante referir que, apesar de existir um número limite de iterações, o programa pode terminar a sua execução mais cedo. Esta situação acontece se ao fim de um dado número de iterações consecutivas, que foi considerado dez, não existir uma variação significativa, que foi estipulado 5% de variação. Estes parâmetros foram escolhidos assim por ser o mais comumente utilizado. Os restantes parâmetros relativos à temperatura e às iterações são depois abordados no seção 4.2.

Passando de seguida para as fases que compõem o SA, começa-se pela função de aceitação de novas soluções, cujo pseudocódigo está apresentado na Figura 12.

```
procedure acceptance_probability (old_cost, new_cost, temperature)
     if new_cost < old_cost then:
          return 1
     else:
          probability ← exp((old_cost - new_cost) / temperature)
          return probability
end acceptance_probability
```

Figura 12 - Pseudocódigo da Fase de Aceitação de Soluções

Esta fase do processo é o ponto diferenciador desta meta-heurística para as restantes, uma vez que permite aceitar soluções piores para explorar mais o espaço de soluções. Caso seja melhor que a solução para já encontrada, a solução é aceite. No caso de ser pior, é efetuado o cálculo da probabilidade de aceitação, cujo valor é depois utilizado noutra fase do processo. Esta probabilidade depende tanto do agravamento do valor objetivo da solução como do parâmetro temperatura. Este último começa num dado valor predefinido, denominado temperatura inicial, e vai diminuindo com o avançar de iterações já realizadas. Isto faz com que a probabilidade de aceitar soluções piores vá decrescendo à medida que se vai chegando ao fim do tempo de execução do método.

A fase da vizinhança também é comum a ambas as versões, e o pseudocódigo encontra-se representado na Figura 13.

```
procedure neighborhood (solution)
        other_solution ← copy of the current_solution but shuffled
        new_solution_1 ← crossover_2_point (current_solution, other_solution)
        new_solution_2 ← apmx (new_solution_1)
        new_solution ← mutate (new_solution_2)
        return new_solution
end neighborhood
```

Figura 13 - Pseudocódigo da Fase de Vizinhança

Nesta fase do procedimento, é utilizado o crossover combinado que consiste em Crossover de dois pontos e APMX (*adapted partial-mapped crossover*). Para além disso, é aplicada uma mutação à nova solução, mais precisamente às máquinas. Neste ponto, é importante referir que apesar de se tratar a solução com os termos *best_solution*, *new_solution*, entre outros, na realidade a solução corresponde a dois vetores. Um dos vetores tem a ordem de execução dos trabalhos e o outro tem a máquina onde cada trabalho é realizado. Na Figura 14 é encontrado um exemplo destes vetores. Por exemplo, através da observação direta, percebe-se que o trabalho 7 é o quinto da sequência e é realizado na máquina 3.

Job Vector	4	1	2	8	7	5	6	3	10	9
Machine Vector	2	2	1	4	3	1	4	2	4	3

Figura 14 – Exemplos de vetores utilizados na solução

O primeiro crossover utilizado é o de dois pontos. Pode ser descrito pelos seguintes passos:

1. Criar um segundo par de vetores com uma ordem aleatória;
2. Selecionar dois pontos aleatórios para cortar os vetores;
3. Trocar os valores em ambos os vetores entre esses pontos;
4. Garantir que não existem trabalhos repetidos nos vetores dos trabalhos, caso haja usar a função *fix_duplicates* criada com o intuito de substituir os duplicados por trabalhos em falta;
5. Retornar os vetores de trabalhos e máquinas.

Na Figura 15, é possível ver o raciocínio inerente a este crossover.

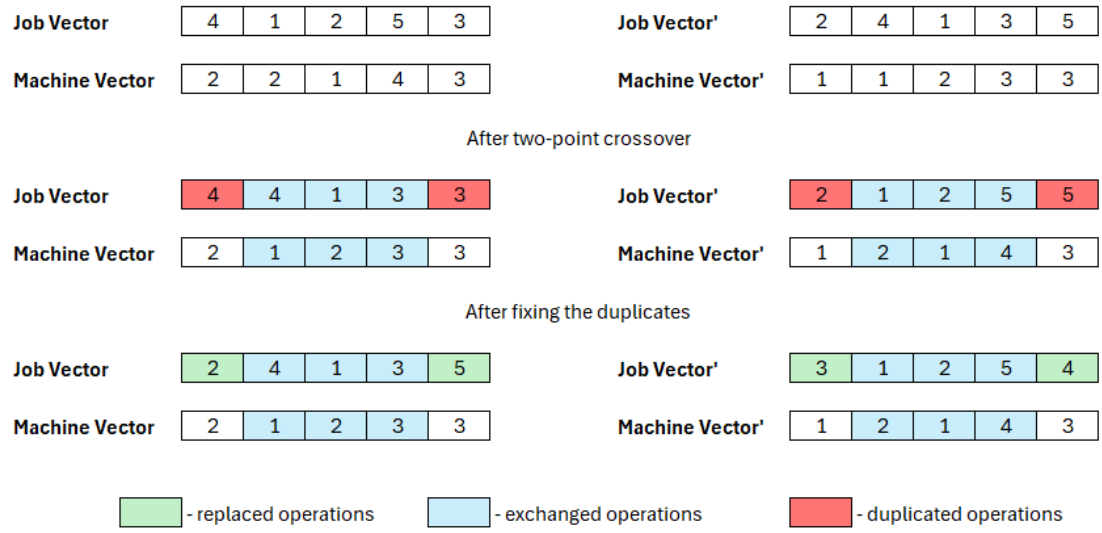


Figura 15 - Crossover de dois pontos

O crossover APMX realiza uma troca parcial entre duas soluções, com base em probabilidades. O funcionamento deste crossover pode ser explicado pelos seguintes pontos:

1. Criar um segundo par de vetores com uma ordem aleatória;
2. Em cada posição do vetor de trabalhos, se o valor aleatório for inferior à probabilidade definida, trocar o trabalho e a máquina com a outra solução;
3. Garantir que não existem trabalhos repetidos nos vetores dos trabalhos, caso haja usar a função *fix_duplicates* criada com o intuito de substituir os duplicados por trabalhos em falta;
4. Retornar os vetores de trabalhos e máquinas.

A Figura 16 exibe um exemplo do funcionamento deste crossover.

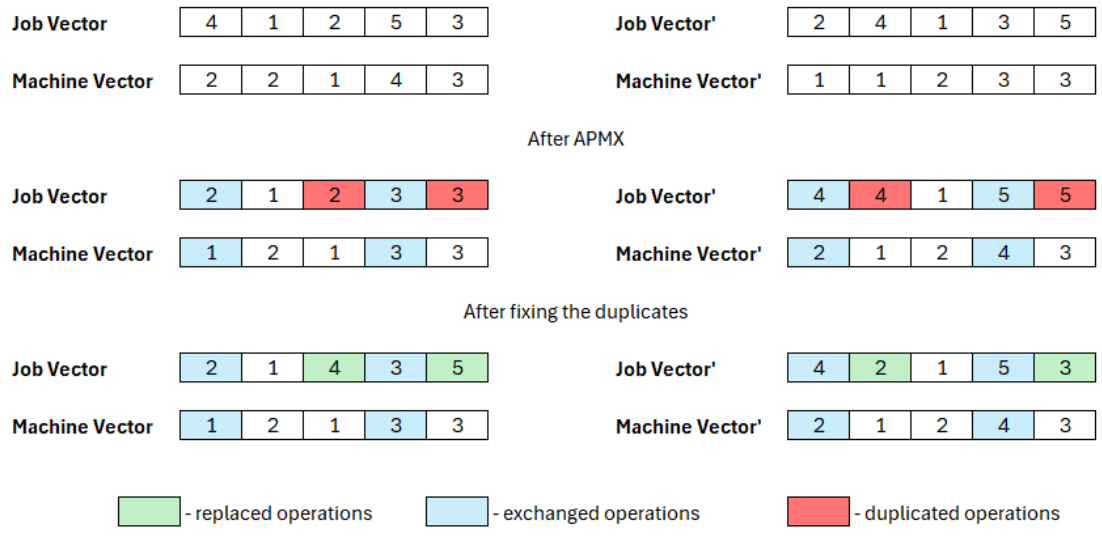


Figura 16 - Crossover APMX

Conforme se pode ver na Figura 16, a primeira e quarta posição sofreram troca, ou seja, o valor aleatório gerado nessas posições foi inferior ao valor da probabilidade definido. Com estas mudanças, ocorreu duplicação de trabalhos no vetor e, por isso, foi necessário recorrer à função *fix_duplicates* para substituir esses trabalhos por aqueles que não estavam incluídos na lista.

A mutação é apenas aplicada ao vetor das máquinas, pois da forma como é feito o crossover, cada trabalho seria sempre realizado na mesma máquina e para testar várias opções não deve ser assim. Portanto, de forma simples, o que é realizado é o seguinte:

1. Criar uma lista com todas as máquinas disponíveis;
2. Selecionar uma posição aleatória no vetor das máquinas;
3. Substituir a máquina dessa posição por uma máquina aleatória da lista de máquinas disponíveis;
4. Retornar o vetor das máquinas.

Na Figura 17, é possível visualizar um exemplo de mutação. Foi escolhida aleatoriamente a quarta posição do vetor para trocar a máquina. De entre as máquinas disponíveis, foi selecionada aleatoriamente a máquina 4 para substituir a 3 naquela posição do vetor.

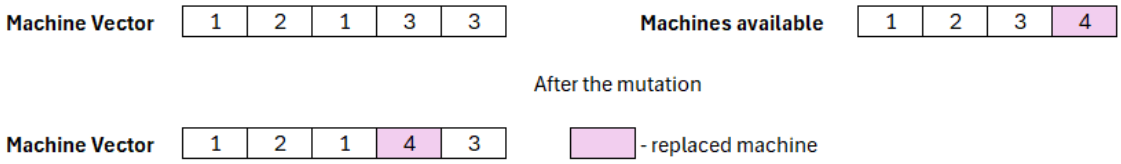


Figura 17 – Mutação

Por fim, resta analisar a forma como é construída a solução a partir dos dois vetores, trabalhos e máquinas. Esta fase também difere entre versões e, portanto, sempre que se chegar a um ponto em que é diferente, essa diferença é registada. A construção da solução pode ser descrita pelos seguintes passos:

- Inicializar matrizes com o estado das máquinas, das ferramentas (e das pessoas, no caso da versão mais complexa);
- Inicializar um vetor que guarda o último instante de funcionamento de cada máquina;
- O vetor dos trabalhos é dividido em dois: job_init_list (primeiro trabalho de cada máquina) e job_remaining_list (trabalhos restantes);
- Começa-se pelo vetor job_init_list porque as condições de início são diferentes. Repetir os passos seguintes para todos os trabalhos dessa lista:
    - Obter posição do trabalho na sequência, a máquina onde se realiza e a duração;
    - Atualizar o tempo inicial da máquina;
    - Registar a alteração de diâmetro (e na segunda versão, a utilização de um operador para essa tarefa);

    - Adicionar as ferramentas necessárias para o trabalho (e na segunda versão, adicionar a utilização de um operador para essa tarefa e contabilizar número de trocas de ferramentas);
    - Na versão 2 do método, caso não haja operadores disponíveis para os *setups* das máquinas, aguarda-se até que exista para depois se efetuar as alterações necessárias;
    - Registar a execução do trabalho no estado da máquina e guardar o instante de tempo em que termina o trabalho.
- Para os restantes trabalhos, presentes no vetor job_remaining_list, é necessário sempre uma comparação com o trabalho anterior para definir *setups*. Repetir os seguintes passos para todos os trabalhos da lista:
    - Obter a posição do trabalho na sequência, a máquina onde se realiza, a duração, o diâmetro e a classificação;
    - Comparar estes parâmetros com os do trabalho anterior executado nessa máquina;
    - Atualizar o tempo de utilização da máquina;
    - Caso haja mudança de diâmetro e/ou de classificação, registar essa alteração e respetivo tempo de *setup* (e na versão 2, adicionar a utilização de um operador para essas alterações);
    - Atualizar as ferramentas, retirando ou inserindo conforme necessário (e na versão 2, adicionar a utilização de um operador para essas alterações e contabilizar todas as trocas feitas);
    - No caso da versão mais complexa, existe sempre a restrição do número de operadores disponíveis e caso não se valide a restrição, a máquina fica em espera até que reúnam as condições necessárias;
    - Registar a realização do trabalho e salvar o instante em que terminou.
- Na vertente de comparação com o modelo, é necessário remover todas as ferramentas de cada máquina no fim do processamento;
- Calcular o valor ou valores objetivo;
- Retornar o valor ou valores objetivo e as matrizes de estado das máquinas, de ferramentas e de pessoas.

Conforme se percebe pela descrição do procedimento, a diferença principal encontra-se na restrição das pessoas que não é considerada na versão de comparação com o modelo. Esta restrição extra faz diferir bastante os resultados, uma vez que dessa forma há esperas nas máquinas quando não há operadores disponíveis.

4.2 Parametrização do Simulated Annealing

Relativamente aos parâmetros que são definidos antes da execução do Simulated Annealing, são os seguintes:

- Temperatura inicial;
- Fator de arrefecimento da temperatura (α);
- Número de iterações limite;
- Número de iterações por temperatura (L).

Um dos fatores mais importantes é o fator de arrefecimento da temperatura ao longo das iterações, o parâmetro L, ou seja, ao fim de quantas iterações a temperatura é diminuída. A sua importância deve-se ao facto de se o arrefecimento for muito rápido, o algoritmo poder ficar preso num mínimo local e caso contrário, o algoritmo poder não convergir para a solução ótima.

Estes parâmetros têm de ser definidos para cada instância, pois variam mediante o tamanho do problema e a sua complexidade. Para a definição destes parâmetros, foram utilizados os cálculos presentes na Figura 18. No caso do exemplo, o objetivo principal era o *makespan*, mas funciona da mesma forma para os outros objetivos.

	Makespan
Pior	112,5
Melhor	82,62

	Temperatura	
Inicial	283,60	+ (Makespan Melhor - Pior) / ln(0,9)
Final	6,49	+ (Makespan Melhor - Pior) / ln(0,01)

α	0,99	
L	0,27	-> Iterações/Reduções
Iterações	100,00	
Reduções	375,00	
T_final	6,54	-> Temp inicial * alpha ^ reduções

Figura 18 - Cálculos para a determinação dos parâmetros

Em primeiro lugar, foi realizado um teste inicial com valores aleatórios, exceto o número de iterações limite que foi estipulado 50. Depois disso, retirou-se o melhor e o pior valor do *makespan* e, a partir desses calculou-se duas temperaturas, a inicial e a final. O valor de α foi arbitrário e considerou-se 0,99 por ser o valor mais comumente utilizado. As iterações dependem do tempo disponível para correr o programa e deve-se considerar nestes cálculos um valor um pouco mais baixo, para que haja tempo de o algoritmo convergir para a solução ótima. As reduções são ajustadas de forma que a T_final se aproxime da temperatura final desejada. O L é obtido através da expressão presente na Figura 18 e devem ser testados vários valores próximos desse, para saber qual permite melhor convergência. Depois é necessário ir ajustando estes parâmetros durante as execuções seguintes, até que estabilizem.

4.3 Heurística construtiva

Foi ainda desenvolvida uma heurística construtiva, por forma a tentar melhorar os resultados obtidos com o modelo matemático e a meta-heurística. Portanto, neste trabalho, a heurística funciona como um complemento aos métodos já aplicados.

Assim, dada uma distribuição de trabalhos pelas máquinas, a solução é construída sequenciando os trabalhos por ordem decrescente do número de ferramentas necessário à realização do trabalho. O que se pretende com isto é colocar no início trabalhos com várias ferramentas e ir ajustando o carrossel conforme as necessidades, o que poderá reduzir as trocas seguintes. Tal como foi referido, é necessário estipular primeiramente quais os trabalhos atribuídos a cada máquina.

A heurística criada utiliza, tal como a meta-heurística, dois vetores como solução: a ordem de execução dos trabalhos e as respetivas máquinas onde se realizam. Para ordenar os trabalhos tendo em conta o número de ferramentas necessárias para cada, é criado um vetor extra que guarda em cada posição o número de ferramentas usadas pelo trabalho colocado naquela posição do vetor dos trabalhos. É depois através deste vetor que se ordena segundo a regra explicada. As alterações de ordem neste vetor refletem-se também nos vetores dos trabalhos e das máquinas, de forma que não se altere a alocação dos trabalhos às máquinas. O esquema presente na Figura 19 retrata exatamente a lógica por trás desta heurística.

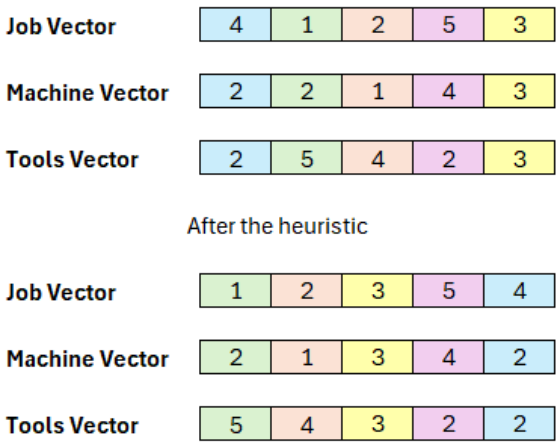


Figura 19 - Funcionamento da heurística construtiva

Portanto, cada coluna de posição dos três vetores funciona como um “set” e quando é feita a ordenação, todo o “set” é deslocado. É isso que se pretende demonstrar com o uso das cores na Figura 19.

A partir do momento que se tem uma nova sequência de trabalhos e máquinas, é calculado o(s) objetivo(s) através do mesmo processo de construção de escalonamento usado na meta-heurística.

5 Resultados Computacionais

Este capítulo dedica-se à exposição dos resultados obtidos e à sua discussão. Depois, é explicada a forma como foram implementados os métodos e em que circunstâncias foram executados os testes com as instâncias geradas. Por fim, são apresentados os resultados de cada abordagem e cenário estudados, tanto individualmente como comparando entre si.

5.1 Geração de instâncias

Tendo por base os dados reais da empresa, foram geradas instâncias de teste. Foram definidos o número de trabalhos, de máquinas disponíveis, de tipos de ferramentas e de operadores disponíveis. Estabeleceu-se que cada operação de *setup* necessita apenas de um operador. Os conjuntos de parâmetros utilizados encontram-se apresentados na Tabela 7.

Tabela 7 - Parâmetros das instâncias de teste

Instância	Máquinas	Trabalhos	Ferramentas	Operadores
Pequena	2	3	4	4
Média	4	14	29	4
Grande_1	10	33	40	4
Grande_2	10	27	32	4
Grande_3	10	27	34	4

5.2 Implementação dos métodos

Os resultados foram obtidos no Microsoft Visual Studio Code 2024 através de programação feita em *Python*. Para o desenvolvimento do modelo matemático, foi utilizada a biblioteca *Pulp* e o solver CPLEX. Os testes computacionais foram realizados num computador com processador Intel (R) Core (TM) i5-9300H CPU @ 2.40GHz com 8.00 GB de memória RAM.

Na Figura 20, é esquematizada a forma de implementação do sistema criado.

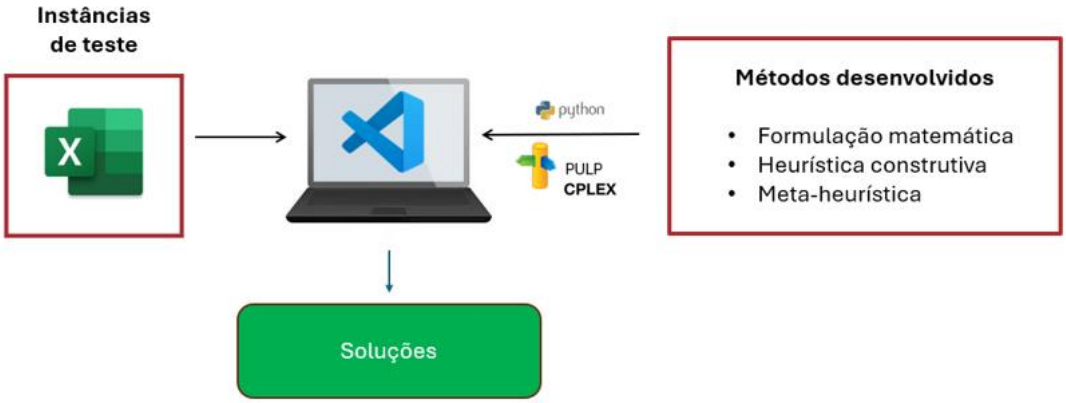


Figura 20 – Implementação dos métodos

5.3 Avaliação do desempenho do Modelo Matemático

Neste subcapítulo, são apresentados os resultados obtidos para os testes feitos com o modelo matemático criado. Apesar de neste tópico as conclusões serem retiradas isoladamente, no subcapítulo 4.3 é comparado o modelo matemático com a meta-heurística desenvolvida sob as mesmas condições.

Na Tabela 8 é mostrada o desempenho do modelo matemático, mas apenas para as instâncias pequena e média. Verifica-se que o GAP obtido é em média 1% para ambas as instâncias, o que é relativamente aceitável.

Tabela 8 - Resultados do modelo matemático

Instância	Makespan (h)			GAP (%)			Tempo (s)		
	Mín	Méd	Máx	Mín	Méd	Máx	Mín	Méd	Máx
Pequena	26,57	26,57	26,57	1,09	1,09	1,09	0,46	0,52	0,66
Média	26,92	26,92	26,92	1,30	1,30	1,30	202	213,60	221

Para analisar melhor a forma como foi construída a solução, analisa-se de seguida a instância média. Os dados para essa instância apresentam-se na Tabela 9.

Tabela 9 - Dados da instância de teste média

ID da ordem	Tempo por peça (h)	Quant. da ordem	Ferramenta 1	Ferramenta 2	Ferramenta 3	Ferramenta 4	Ferramenta 5	Diâmetro	Categoria do Material
1	0,18	22	TOOL.5	TOOL.1	TOOL.8	TOOL.9	TOOL.10	7	1
2	0,14	107	TOOL.1	TOOL.2	TOOL.3	-	-	4	2
3	0,1	15	TOOL.1	TOOL.3	0	-	-	5	3
4	0,15	172	TOOL.4	TOOL.1	TOOL.2	-	-	5	4
5	0,15	107	TOOL.4	TOOL.1	TOOL.2	-	-	4	4
6	0,16	12	TOOL.1	TOOL.6	TOOL.7	-	-	4	5
7	0,16	8	TOOL.1	TOOL.6	TOOL.7	-	-	5	5
8	0,22	7	TOOL.22	TOOL.23	TOOL.24	TOOL.25	-	4	6
9	0,14	7	TOOL.23	TOOL.25	TOOL.11	TOOL.13	-	4	7
10	0,12	108	TOOL.11	TOOL.12	TOOL.13	TOOL.26	-	4	8
11	0,11	22	TOOL.20	TOOL.3	TOOL.27	TOOL.28	-	5	9
12	0,11	22	TOOL.20	TOOL.5	TOOL.3	TOOL.21	-	4	10
13	0,14	54	TOOL.14	TOOL.17	TOOL.18	TOOL.15	-	7	11
14	0,12	12	TOOL.8	TOOL.16	TOOL.19	TOOL.19	-	7	12

Como é possível observar na Figura 21, os tempos de conclusão de cada máquina foram muito semelhantes entre si, todos a rondar as 26 horas. O trabalho 4, devido ao elevado tempo de processamento, foi o único atribuído à máquina 2. Para uma melhor interpretação dos resultados, é importante clarificar que o Nr.Z corresponde ao número de montagens de ferramentas e o Nr.X às desmontagens. Estas designações vêm das variáveis de decisão do modelo matemático desenvolvido.

Máquina	Posição	Ordem	Tempo Processamento	Nr.Z	Nr.X	Tempo Trocas	Diâmetro	Setup	Tempo Total	Makespan
1	1	4	25,8	3	0	0,18	0,41	0	26,39	26,57
				0	3	0,18			0,18	

Máquina	Posição	Ordem	Tempo Processamento	Nr.Z	Nr.X	Tempo Trocas	Diâmetro	Setup	Tempo Total	Makespan
2	1	12	2,42	4	0	0,24	0,41	0	3,07	26,62
2	2	8	1,54	4	4	0,48	0	0,44	2,46	
2	3	2	14,98	3	4	0,42	0	0,28	15,68	
2	4	11	2,42	3	2	0,3	0,41	0,22	3,35	
2	5	3	1,5	1	3	0,24	0	0,2	1,94	
				0	2	0,12			0,12	

Máquina	Posição	Ordem	Tempo Processamento	Nr.Z	Nr.X	Tempo Trocas	Diâmetro	Setup	Tempo Total	Makespan
3	1	10	12,96	4	0	0,24	0,41	0	13,61	**26,92**
3	2	6	1,92	3	4	0,42	0	0,32	2,66	
3	3	9	0,98	4	3	0,42	0	0,28	1,68	
3	4	13	7,56	4	4	0,48	0,41	0,28	8,73	
				0	4	0,24			0,24	

Máquina	Posição	Ordem	Tempo Processamento	Nr.Z	Nr.X	Tempo Trocas	Diâmetro	Setup	Tempo Total	Makespan
4	1	1	3,96	5	0	0,3	0,41	0	4,67	26,79
4	2	5	16,05	2	4	0,36	0,41	0,3	17,12	
4	3	7	1,28	2	2	0,24	0,41	0,32	2,25	
4	4	14	1,44	4	3	0,42	0,41	0,24	2,51	
					4	0,24			0,24	

Figura 21 - Resultado da instância de teste média através do modelo matemático

A Figura 22 representa a ordem com que os trabalhos foram realizados em cada máquina e a Figura 23 demonstra a utilização das ferramentas. Ambos os gráficos são indexados à posição.

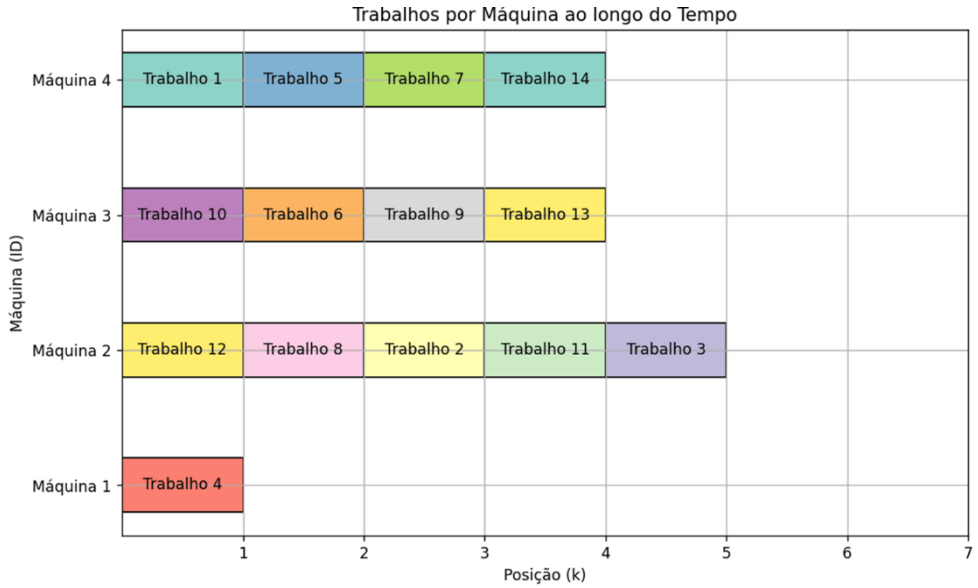


Figura 22 - Sequência dos trabalhos por máquina da Instância de teste média

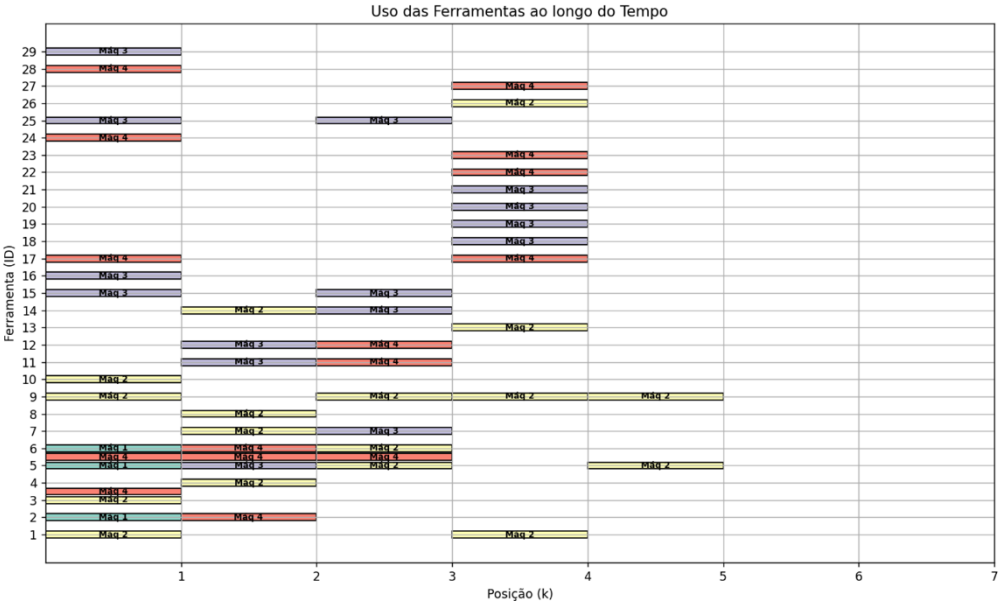


Figura 23 - Distribuição das ferramentas por posição nas máquinas da Instância média

Contudo, estando indexado à posição, não dá uma visão realista do escalonamento em termos temporais. Assim, foi construído o gráfico de *Gantt* presente na Figura 24, indexado ao tempo, o que já transmite que apesar de só existir um trabalho na máquina 2, em termos de tempo, esta está equiparada às outras máquinas.

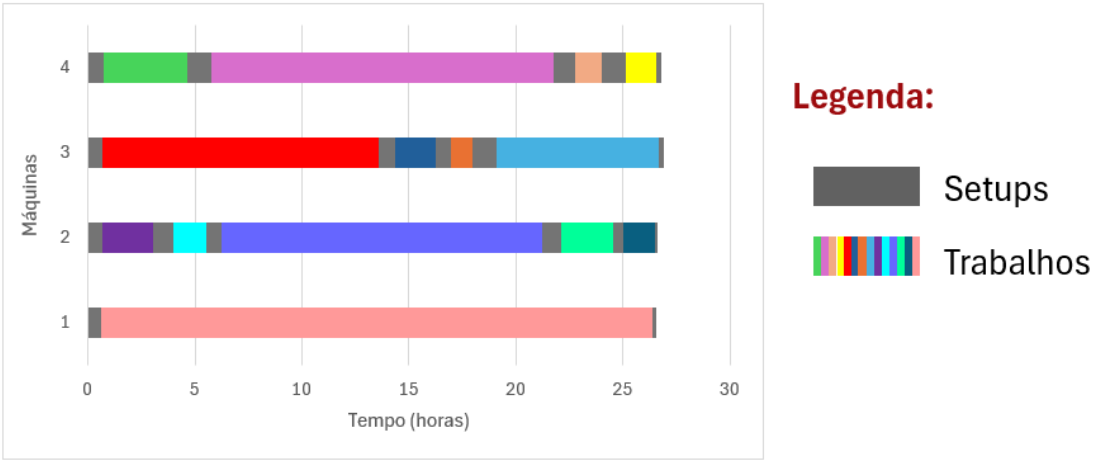


Figura 24 - Gráfico de Gantt da Instância de teste média

Para as instâncias grandes, o modelo matemático não conseguiu encontrar solução. Para tentar perceber a razão por detrás disso, foi feita uma análise ao número de variáveis necessárias para cada instância de teste, visível na Tabela 10.

Tabela 10 - Comparação do número de variáveis necessárias para cada instância

Instâncias	W_{jkm}	Y_{tkm}	Z_{tkm}	X_{tkm}	H_{jkm}	C_{jkm}	Δ_m	Total
Pequena	18	24	24	24	18	18	2	128
Média	784	1624	1624	1624	784	784	4	7228
Grande_1	10890	13860	13860	13860	10890	10890	10	74260
Grande_2	7290	8640	8640	8640	7290	7290	10	47800
Grande_3	7290	9720	9720	9720	7290	7290	10	51040

Conforme é possível observar na Tabela 10, passar de uma instância média para grande, aumenta exponencialmente o número de variáveis necessárias e, por isso, o problema torna-se muito complexo e exige um esforço computacional muito superior.

O modelo, tal como foi apresentado anteriormente na seção 3.3, só consegue encontrar solução válida em tempo útil para as instâncias pequena e média. Por isso, foi aplicado um modelo matemático relaxado às restantes instâncias grandes. O facto de ser relaxado significa que algumas condições são "relaxadas", para tornar o problema mais simples de resolver. Neste caso, as variáveis de decisão $W_{jkm}, Y_{tkm}, Z_{tkm}, X_{tkm}, H_{jkm}$ *e* C_{jkm} deixaram de ser binárias e passaram a ser contínuas, com *lower bound* (limite inferior) 0 e *upper bound* (limite superior) 1. Portanto, o *lower bound* é o valor mínimo que a variável pode atingir e o *upper bound* é o maior valor possível para a variável.

Este modelo matemático relaxado permite apenas gerar uma atribuição dos trabalhos às máquinas, ficando a faltar a questão da sequenciação e das trocas de ferramentas. Sendo uma

versão mais simplificada, o programa demora em média apenas 62 segundos para encontrar uma solução. Esta atribuição dos trabalhos às máquinas vai ser depois utilizada no tópico 5.7.

5.4 Comparação do Modelo Matemático com a meta-heurística

Tal como concluído no subcapítulo anterior, o modelo matemático desenvolvido não consegue fornecer solução em tempo útil para instâncias de maiores dimensões. Por isso, foi implementada uma meta-heurística, o Simulated Annealing, para comparar o desempenho de ambas as abordagens. Para esta comparação ser feita em condições de igualdade com o modelo matemático, é utilizada a primeira versão criada, referida no tópico 3.4.1, que não contém restrições extra e que tem como objetivo minimizar apenas o *makespan*.

Na Tabela 11, é possível ver os resultados obtidos com ambas as abordagens, para as instâncias de teste. São referidos os valores mínimos encontrados.

Tabela 11 - Comparação entre o modelo matemático e a meta-heurística

Instâncias	Modelo Matemático		Meta-heurística	
	Makespan (h)	Tempo (s)	Makespan (h)	Tempo (s)
Pequena	26,57	0,46	26,57	196
Média	26,92	202	28,59	964
Grande_1	-	-	62,89	1021
Grande_2	-	-	81,77	835
Grande_3	-	-	81,77	195

Tal como é possível verificar na Tabela 11, para a instância pequena consegue-se atingir o mesmo *makespan* com ambos os métodos. Já na instância média, o *makespan* encontrado pelo modelo matemático é melhor que o encontrado com a meta-heurística. Os tempos computacionais da meta-heurística são sempre maiores, uma vez que são realizadas várias iterações à procura da melhor solução possível. Aliás, neste tipo de meta-heurística, Simulated Annealing, o pretendido é que à medida que a temperatura diminui, se vá caminhando para a solução ótima, ou seja, algo como o que se verifica, por exemplo, na Figura 25.

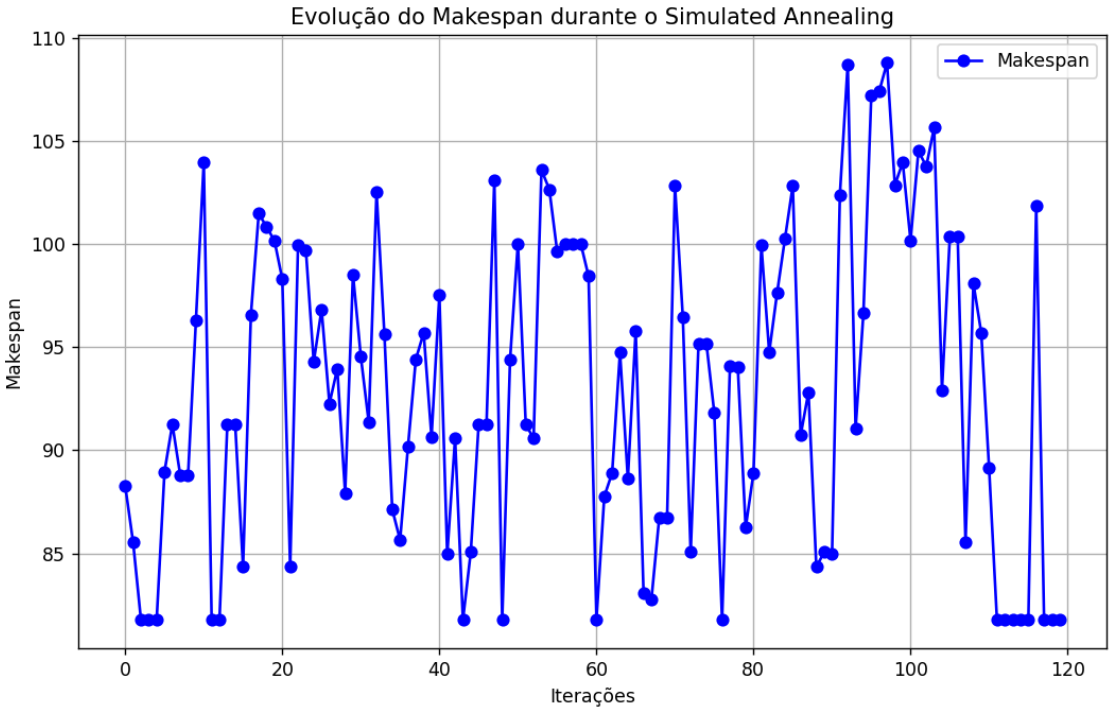


Figura 25 - Evolução do Makespan ao longo das iterações

Tal como se consegue verificar no gráfico de tendência da Figura 25, perto das 110 iterações, o modelo começa a convergir para uma solução ótima. É normal e expectável que isto aconteça, uma vez que reduzindo a temperatura, vai sendo cada vez menor a probabilidade de aceitar soluções piores.

Na Tabela 12, apresenta-se o desempenho isolado da primeira e mais simplificada versão da meta-heurística. Assim, foi possível obter soluções para todas as instâncias em relativamente pouco tempo computacional.

Tabela 12 - Resultados da 1ª versão da Meta-heurística

Instâncias	Makespan (h)			Número de iterações			Tempo (s)		
	Mín	Méd	Máx	Mín	Méd	Máx	Mín	Méd	Máx
Pequena	26,57	26,57	26,57	279	329	368	196	286,20	442
Média	28,59	32,35	35,38	150	150	150	694	718,60	750
Grande_1	62,89	70,54	74,01	100	100	100	1021	1212	1792
Grande_2	81,77	81,77	81,77	100	100	100	835	847,70	866
Grande_3	81,77	81,77	81,77	25	88	120	195	1387	3038

5.5 Desempenho do Simulated Annealing

Neste subcapítulo, são analisados os vários cenários realizados com a segunda versão da meta-heurística. Esta difere da primeira pelo facto de incluir a restrição dos recursos humanos, o que

pode implicar que a máquina fique parada em alguns instantes de tempo à espera de que algum dos operadores esteja disponível para fazer os *setups*. Para além disso, apesar de alguns cenários avaliarem apenas um objetivo, existem outros casos em que se avaliam dois objetivos. Nestes últimos casos, há ainda a divisão em objetivos com importâncias iguais (multiobjectivo) ou diferentes. Para cada cenário, foram feitos 5 testes em cada instância.

Portanto, na Tabela 13, estão enumerados os vários cenários estudados com a meta-heurística Simulated Annealing, ou seja, quais os parâmetros a minimizar.

Tabela 13 - Cenários avaliados com a meta-heurística

Cenário	Objetivo(s) de minimização
Cenário_1	1. *Makespan* 2. Número de trocas de ferramentas
Cenário_2	Número de trocas de ferramentas
Cenário_3	1. Número de trocas de ferramentas 2. Número de máquinas utilizadas
Cenário_4	1. Número de máquinas utilizadas 2. Número de trocas de ferramentas
Cenário_5	• Número de trocas de ferramentas • Número de máquinas utilizadas

Como se pode ver na Tabela 13, os cenários 1, 3 e 4 têm dois objetivos cada, mas com importâncias diferentes, isto é, um dos objetivos tem mais relevância que o outro. O cenário 5 também tem dois objetivos, mas estes são igualmente relevantes. Já o cenário 2 apenas tem como objetivo minimizar o número de trocas de ferramentas.

Avaliando primeiramente os cenários individualmente e seguindo a ordem com que foram apresentados, no Cenário_1 o objetivo principal era minimizar o *makespan* e, depois, minimizar ainda o número de trocas de ferramentas realizadas ao longo de todo o processo.

Tabela 14 - Resultados do Cenário_1

Instância	Makespan (h)			Número de trocas		
	Mín	Méd	Máx	Mín	Méd	Máx
Pequena	26,39	26,39	26,39	6	6	6
Média	30,85	31,19	31,80	65	65	66
Grande_1	66,14	71,86	76,01	157	164	167
Grande_2	81,59	81,59	81,59	112	117	121
Grande_3	81,59	81,59	81,60	103	108	117

Tal como é possível verificar pela observação da Tabela 14, existe evidência de estabilidade dos resultados, uma vez que a variação entre os valores máximos e mínimos não é muito significativa. Já pela análise da Tabela 15, percebe-se que na instância pequena o número de iterações é muito variável, o que pode ser explicado pelo facto de existir um critério de paragem

que interrompe a execução quando não há melhoria significativa do objetivo ao fim de um dado número de iterações. Quando o algoritmo não encontra nenhuma solução melhor após várias iterações, isso pode significar que este está preso numa determinada região, por exemplo, num ótimo local e, portanto, continuar a execução resulta em tempo computacional desnecessário.

Tabela 15 - Métricas do Cenário_1

Instância	Número de iterações			Tempo (s)		
	Mín	Méd	Máx	Mín	Méd	Máx
Pequena	259	318	369	98	125,30	144
Média	150	150	150	563	592	636
Grande_1	100	100	100	866	888,33	906
Grande_2	150	150	150	971	991,30	1029
Grande_3	120	120	120	675	700	730

O Cenário_2 tem apenas um objetivo, minimizar o número de trocas de ferramentas. Na Tabela 16 são apresentados os resultados conseguidos neste cenário.

Tabela 16 - Resultados e Métricas do Cenário_2

Instância	Número de trocas			Número de iterações			Tempo (s)		
	Mín	Méd	Máx	Mín	Méd	Máx	Mín	Méd	Máx
Pequena	4	4	4	309	352	414	111	127,30	150
Média	54	55	56	150	150	150	550	577	593
Grande_1	140	142	146	100	100	100	1400	2167,30	3434
Grande_2	94	97	99	110	110	110	641	795,70	1090
Grande_3	89	91	92	120	120	120	710	872	1193

Assim, neste cenário, semelhante ao que aconteceu com o anterior, os resultados são bastante estáveis, exibindo pouca variação entre o valor mínimo e o valor máximo registados. As instâncias grandes que refletem a realidade de um planeamento semanal nesta indústria têm tempos computacionais numa média de 1278 segundos (cerca de 21 minutos), o que é bastante aceitável.

Com o Cenário_3 regressa a função multiobjectivo e, tal como no Cenário_1, os objetivos têm importâncias diferentes. O número de trocas é o principal objetivo a minimizar e, em segundo plano, o número de máquinas. Na Tabela 17, apresentam-se os resultados atingidos neste cenário.

Tabela 17 - Resultados do Cenário_3

Instância	Número de trocas			Número de máquinas		
	Mín	Méd	Máx	Mín	Méd	Máx
Pequena	4	4	4	1	1	1
Média	54	54	54	4	4	4
Grande_1	130	138	147	8	8	9
Grande_2	94	97	100	6	7	10
Grande_3	92	92	93	7	8	8

Tal como se pode concluir pela observação da Tabela 17, os resultados obtidos para as instâncias pequena e média não apresentam nenhuma variação. Em todos os 5 testes realizados, os resultados foram os mesmos. Esta estabilidade tão acentuada dos valores pode ser justificada pela menor quantidade de dados que estas instâncias apresentam e, também pelo grande número de iterações feitas ao longo do processo. Como a quantidade de ordens é significativamente mais pequena que as instâncias grandes, as possibilidades de sequência e de atribuição de trabalhos às máquinas são menos e, portanto, realizando várias iterações e em pouco tempo, consegue-se explorar muitas dessas opções.

Tabela 18 - Métricas do Cenário_3

Instância	Número de iterações			Tempo (s)		
	Mín	Méd	Máx	Mín	Méd	Máx
Pequena	589	594	598	212	214	216
Média	150	150	150	578	580,30	584
Grande_1	100	100	100	834	1127,67	1670
Grande_2	150	150	150	664	850,30	954
Grande_3	120	120	120	681	683,30	688

A principal conclusão a tirar da Tabela 18 é a elevada diferença entre os tempos computacionais e o número de iterações das instâncias. Quanto maior for a quantidade de dados na instância, mais tempo demora a executar o programa para uma iteração e, por isso, menor número de iterações testadas e maior tempo computacional. Em jeito de exemplo, comparando os valores máximos da instância pequena e da Grande_1, a diferença é de 4 minutos para 28 minutos.

O Cenário_4 avalia exatamente os mesmos objetivos que o Cenário_3, mas com a ordem inversa de importância. Portanto, neste cenário, o objetivo principal é minimizar o número de máquinas usadas e, depois, o número de trocas de ferramentas. Algumas empresas podem ter este objetivo de reduzirem o número de máquinas usadas se, por exemplo, o plano que têm de produção seja possível executar na semana de trabalho sem terem de ligar todas as máquinas e poupando algumas configurações iniciais.

A Tabela 19 corresponde aos resultados obtidos com este cenário.

Tabela 19 - Resultados do Cenário_4

Instância	Número de máquinas			Número de trocas		
	Mín	Méd	Máx	Mín	Méd	Máx
Pequena	1	1	1	4	4	4
Média	2	2	2	68	69	70
Grande_1	6	6	6	152	158	170
Grande_2	4	5	6	112	115	130
Grande_3	4	5	5	100	105	113

Analisando os resultados presentes na Tabela 19, percebe-se que alterar a ordem de importância dos objetivos tem um grande impacto nos valores. O número de máquinas, ao passar para objetivo mais relevante, diminuiu de 7 para 4 máquinas utilizadas na instância Grande_3. Por outro lado, o objetivo que passou para segundo plano, piorou em relação ao Cenário_3, passando de 92 para 100 trocas de ferramentas.

Tabela 20 - Métricas do Cenário_4

Instância	Número de iterações			Tempo (s)		
	Mín	Méd	Máx	Mín	Méd	Máx
Pequena	602	620	644	224	228	235
Média	150	150	150	553	579	596
Grande_1	100	100	100	816	828,70	844
Grande_2	150	150	150	941	948	956
Grande_3	120	120	120	688	696,30	706

A Tabela 20 apresenta as métricas registadas com o Cenário_4 para todas as instâncias de teste. Comparativamente com o cenário anterior, percebe-se que esta troca de ordem de importância dos objetivos não teve grande impacto nos tempos computacionais, uma vez que estes são semelhantes aos registados na Tabela 18.

Por último, resta analisar individualmente o Cenário_5. Este cenário é completamente diferente dos restantes, uma vez que os dois objetivos avaliados têm exatamente a mesma importância no problema, ou seja, uma solução só é melhor se ambos os valores objetivos forem melhores ou iguais aos encontrados até ao momento. Na Tabela 21, encontram-se os resultados obtidos.

Tabela 21 - Resultados do Cenário_5

Instância	Número de máquinas			Número de trocas		
	Mín	Méd	Máx	Mín	Méd	Máx
Pequena	1	1	1	4	4	4
Média	3	4	4	58	58	58
Grande_1	7	7	8	143	145	148
Grande_2	6	6	7	96	97	98
Grande_3	6	7	9	86	88	91

Ao observar os resultados presentes na Tabela 21, conclui-se que com esta abordagem há um maior compromisso entre os valores de ambos os objetivos. Isto quer dizer que os resultados dos objetivos individualmente podem ser piores quando comparados com os cenários de importâncias diferentes, mas são melhores quando analisados na globalidade. Existe ainda um caso em que se conseguiu atingir um valor menor no número de trocas, para a instância Grande_3, uma vez que o melhor valor encontrado nos outros dois cenários foi de 92 trocas e neste cenário, conseguiu-se reduzir para 86 trocas de ferramentas.

Tabela 22 - Métricas do Cenário_5

Instância	Número de iterações			Tempo (s)		
	Mín	Méd	Máx	Mín	Méd	Máx
Pequena	597	600	603	218	221,50	225
Média	150	150	150	583	588,50	594
Grande_1	100	100	100	955	1010,30	1099
Grande_2	150	150	150	933	1215	1649
Grande_3	120	120	120	701	735	764

Já em relação às métricas registadas na Tabela 22, nota-se que no caso do número de iterações executadas e no tempo computacional utilizado, a variação não é significativa, isto é, os valores são muito semelhantes aos encontrados nos restantes cenários.

5.6 Comparação dos cenários da meta-heurística

Neste subcapítulo são comparados os vários cenários realizados com a meta-heurística, já mencionados no subcapítulo anterior.

Em primeiro lugar, compara-se o número de trocas de ferramentas em todos os cenários, de forma a perceber o impacto nesta métrica do(s) objetivo(s) utilizados. Na Tabela 23, apresenta-se um quadro resumo com os melhores valores encontrados em cada cenário estudado.

Tabela 23 - Comparação do número de trocas de ferramentas nos vários cenários

Instância	Cenário_1	Cenário_2	Cenário_3	Cenário_4	Cenário_5
Pequena	6	4	4	4	4
Média	65	54	54	68	58
Grande_1	157	140	130	152	143
Grande_2	112	94	94	112	96
Grande_3	103	89	92	100	86

Foram sinalizados a verde os melhores resultados conseguidos para cada instância de teste. Conforme se pode concluir para este objetivo em específico, minimizar o número de trocas, o melhor cenário para a maioria das instâncias foi o Cenário_3 que conta com dois objetivos, mas minimizar o número de trocas é o mais importante. Portanto, estas seriam as conclusões esperadas, dada a maior relevância dada a esta métrica, em relação aos restantes cenários.

Em seguida, compara-se o número de máquinas utilizadas, nos três cenários em que foi um dos objetivos. Essa comparação está representada no quadro resumo da Tabela 24.

Tabela 24 - Comparação do número de máquinas nos vários cenários

Instância	Cenário_3	Cenário_4	Cenário_5
Pequena	1	1	1
Média	4	2	3
Grande_1	8	6	7
Grande_2	6	4	6
Grande_3	7	4	6

Tal como se pode verificar na Tabela 24, mais uma vez, o cenário onde se registam os melhores valores é aquele que prioriza este objetivo em detrimento dos restantes. Neste caso, o Cenário_4 é o que apresenta valores menores para o número de máquinas utilizadas no processo. A instância pequena, tal como aconteceu na métrica anterior, acaba por ter quase sempre o mesmo valor independentemente do cenário, mas isso pode ser explicado pela menor quantidade de dados e, por isso, menos sequências e atribuições possíveis para analisar.

Por último, uma análise extra que compara o *makespan* no Cenário_1 onde é o objetivo principal, com o Cenário_4, onde não é sequer objetivo. Essa comparação está representada na Tabela 25.

Tabela 25 - Comparação do makespan em dois cenários diferentes

Instância	Cenário_1	Cenário_4
Pequena	26,39	44,25
Média	30,85	67,17
Grande_1	66,14	93,79
Grande_2	81,59	106,10
Grande_3	81,59	88,54

Tal como se pode comprovar pela observação da Tabela 25, os resultados foram sempre melhores no Cenário_1 em que o *makespan* é objetivo de minimização. Contudo, não é esse o ponto de análise pretendido com esta comparação. O Cenário_4 tem como maior objetivo minimizar o número de máquinas, e esta discrepância grande entre valores leva a crer que estas duas métricas, *makespan* e número de máquinas utilizadas, são inversamente relacionadas. Isto pode ser explicado pelo facto de ao usar menos máquinas, as restantes terem de executar mais carga de trabalhos e, portanto, demorarem mais tempo a terminar todo o processo. Por esta razão, é muito importante as empresas definirem bem e conscientemente os seus objetivos, pois existem objetivos que acabam por não ser muito compatíveis com os outros, como este caso.

5.7 Desemprenho da heurística construtiva

A heurística construtiva foi desenvolvida com vista a melhorar as soluções da meta-heurística e do modelo matemático para as instâncias de maior dimensão. Neste subcapítulo, será dada continuidade à solução gerada para as instâncias grandes, do modelo matemático relaxado. Contudo, o modelo relaxado não gera soluções admissíveis porque o relaxamento de algumas condições, nomeadamente das variáveis binárias passarem para inteiras entre 0 e 1, leva a que não seja possível extrair uma sequência de trabalhos em cada máquina. Assim, este modelo apenas consegue atribuir os trabalhos às máquinas.

Em primeiro lugar, foi aplicado o modelo matemático relaxado às três instâncias de teste grandes. Em média, o tempo computacional necessário para correr o programa foi de aproximadamente 1 minuto. Depois, foi aplicada a meta-heurística, neste caso, o Cenário_5, tendo em conta a atribuição dos trabalhos às máquinas indicada pelo modelo e usando a sequência em ordem crescente do número da ordem de produção. A meta-heurística deu uma solução válida, com sequências de trabalhos em cada máquina. Por fim, estes vetores solução foram utilizados na heurística construtiva, para desenhar uma nova solução e comparar com o resultado da meta-heurística. Na Tabela 26, apresenta-se um resumo dos resultados obtidos em cada fase distinta.

Tabela 26 - Resultados Heurística Construtiva

Instância	Número de máquinas	Número de trocas	
Grande_1	7	143	Cenário_4
Grande_2	6	96	
Grande_3	6	86	
Grande_1	9	125	Meta-heurística
Grande_2	8	87	
Grande_3	10	89	
Grande_1	9	119	Heurística construtiva
Grande_2	8	91	
Grande_3	10	91	

Portanto, na primeira fila da Tabela 26, é possível encontrar os melhores resultados encontrados com o Cenário_4 já apresentado. Na segunda fila, estão os resultados do mesmo cenário aplicado partindo não de uma solução aleatória, mas sim do resultado do modelo matemático relaxado. A última fila tem os resultados da heurística construtiva aplicada com base nos resultados da segunda fila. Observando os valores, percebe-se que em relação ao objetivo do número de máquinas utilizadas, o modelo relaxado e a heurística não conseguiram melhorar as soluções já encontradas. Em relação ao número de trocas, os resultados foram mais dispersos, uma vez que cada instância registou o melhor valor com abordagens diferentes.

Dado que a heurística construtiva só conseguiu melhorar um resultado, conclui-se que começar por fazer os trabalhos com maior número de ferramentas requeridas não é necessariamente melhor, isto é, depende de caso para caso.

6 Conclusão

A crescente competitividade tem estimulado as empresas a melhorar a eficiência dos seus processos produtivos, tornando a flexibilidade e a adaptabilidade essenciais para um sistema produtivo eficaz. Nesse contexto, as máquinas, que são a base de qualquer sistema produtivo, evoluíram com a tecnologia para desempenhar múltiplas tarefas, reduzindo a necessidade de diversos equipamentos. No entanto, essas máquinas têm uma capacidade limitada de armazenamento de ferramentas no carrossel, o que exige trocas frequentes para lidar com a variedade de tarefas e produtos.

Nesse sentido, este estudo aborda o problema de escalonamento de trabalhos com a troca de ferramentas nas máquinas, conhecido como Job Sequencing and Tool Switching Problem (SSP). A resolução deste problema aborda pontos cruciais da realidade das empresas, como setups dependentes da sequência, configurações iniciais das máquinas e operadores disponíveis.

Este trabalho pode dividir-se em dois grandes momentos de investigação: a elaboração de um novo modelo matemático, mais atual e mais direcionado para o problema em questão da indústria metalomecânica; e o estudo do impacto da aplicação de heurística e meta-heurística nos resultados obtidos.

Relativamente ao modelo de programação linear inteira mista, identificam-se 3 contribuições principais. O primeiro ponto é a questão das configurações iniciais das máquinas. Na literatura, é comum encontrar modelos com o pressuposto das máquinas começarem prontas para produção. Contudo, isso não reflete a realidade empresarial pois esses tempos de preparação, antes de começar a produção em si, existem sempre e devem ser contabilizados no processo de planeamento e escalonamento de trabalhos. Além disso, a incorporação da etapa de remoção de ferramentas torna o problema mais complexo e realista, uma vez que não basta contabilizar montagens de ferramentas , as remoções também utilizam tempo e recursos e, por isso, devem ser tidas em conta. Por último, os setus dependentes da sequência, tanto as trocas de ferramentas já mencionadas como outros ajustes necessários entre trabalhos. No caso concreto deste problema, incluem-se nos setups a alteração do diâmetro da haste da máquina conforme o diâmetro da peça a produzir e a mudança de categoria do produto em fabrico. O

modelo desenvolvido tem como objetivo único minimizar o makespan, isto é, o tempo de conclusão de todos os trabalhos.

No que diz respeito à meta-heurística, foi selecionado o Simulated Annealing para a resolução deste problema. Foram desenvolvidas duas versões da meta-heurística, a primeira considerando as mesmas características que o modelo matemático, para facilitar a comparação entre abordagens e a segunda versão com restrições adicionais. Portanto, esta segunda versão inclui a disponibilidade de operadores, aspecto não considerado no modelo matemático. Isto quer dizer que não havendo operadores disponíveis para a realização de um setup, a máquina fica parada em espera, e estes tempos são contabilizados no tempo final. De forma a perceber o impacto do objetivo utilizado nos resultados, foram estudados vários cenários, quer com apenas um objetivo, quer com dois objetivos e, neste último caso, com importâncias diferentes ou iguais. Entre os objetivos analisados encontram-se o makespan, o número de trocas de ferramentas e o número de máquinas utilizadas.

Foi, ainda, aplicada uma heurística construtiva na tentativa de melhorar as soluções já encontradas com os outros métodos. A heurística baseia-se numa regra de ordenação de trabalhos, por ordem decrescente do número de ferramentas requerido para a realização do trabalho. O método desenvolvido necessita de um input, da atribuição dos trabalhos às máquinas, uma vez que este só se dedica à sequenciação dos trabalhos. Depois, a construção do escalonamento é feita de forma semelhante à meta-heurística já referida.

Relativamente aos resultados conseguidos com as várias abordagens, o modelo matemático apenas conseguiu gerar soluções válidas e admissíveis para as instâncias pequena e média, tendo sido necessário "relaxar" o modelo para conseguir obter apenas a atribuição dos trabalhos às máquinas. Tanto a meta-heurística como a heurística construtiva fornecem soluções válidas para todas as instâncias de teste. Portanto, pode-se concluir que os modelos matemáticos podem não ser a melhor opção para problemas de grandes dimensões devido ao aumento exponencial de variáveis geradas, que faz com que o processo seja muito moroso. Os resultados obtidos foram estáveis, o que indica que os métodos estavam bem parametrizados. De um modo geral, a meta-heurística conseguiu os melhores resultados, para os vários objetivos, mas o modelo matemático matemático, para as instâncias mais pequenas permite obter melhores soluções em menor tempo computacional.

Em relação aos tempos computacionais, o modelo matemático demonstrou que é capaz de responder rapidamente, com tempos até cerca de 2 minutos. O Simulated Annealing mostrou conseguir gerar boas soluções até para instâncias grandes e com tempos muito razoáveis de cerca de 15 minutos para as instâncias grandes (neste caso, com até 10 máquinas, 33 trabalhos e 40 ferramentas). A heurística construtiva, consegue soluções admissíveis para instâncias grandes em apenas 1 minuto. Contudo, tem como desvantagem necessitar da distribuição dos trabalhos pelas máquinas como entrada de informação.

6.1 Limitações e investigação futura

Os trabalhos futuros deverão focar-se principalmente na melhoria da meta-heurística desenvolvida, o Simulated Annealing, uma vez que é um método com bom desempenho neste tipo de problemas de escalonamento. Existe margem para melhoria, dado que no tempo disponível para o projeto, não foram possíveis executar todas as ideias. Também pode ser estudado com mais pormenor o impacto de utilizar o modelo matemático ou a heurística construtiva como solução inicial da meta-heurística e perceber qual das abordagens tem melhores resultados. A opção utilizada neste trabalho de começar com uma solução aleatória também pode ser comparada com as outras formas de solução inicial já referidas. Poderá ser proposta, ainda, uma heurística construtuiva/meta-heurística que faça o escalonamento dos trabalhos com base em famílias de produtos, ou seja, que sejam agrupados os trabalhos em famílias com características semelhantes, como a categoria e o diâmetro das peças, e que a atribuição dos trabalhos às máquinas e a sua sequenciação seja feita com base nisso.

Os trabalhos de investigação são normalmente muito complexos e exigentes, sendo a principal dificuldade o facto de ser inconstante, ou seja, de estar constantemente em mudança à procura da melhor abordagem possível. Acaba por ser também um trabalho de tentativa-erro, uma vez que é necessário experimentar para perceber se é para mudar o rumo da pesquisa ou não. Devido ao tamanho e complexidade do problema em estudo, a maior dificuldade prendeu-se com o desenvolvimento do modelo matemático e a determinação de todas as restrições necessárias por forma a retratar o melhor possível as características do problema.

Referências

Ahmadi, E., Goldengorin, B., Süer, G. A., & Mosadegh, H. 2018. A hybrid method of 2-TSP and novel learning-based GA for job sequencing and tool switching problem. *Applied Soft Computing, 65*, 214-229.

Allahverdi, A., & Soroush, H. M. 2008. The significance of reducing setup times/setup costs. *European Journal of Operational Research*, *187*(3), 978-984.

Allahverdi, A. 2015. The third comprehensive survey on scheduling problems with setup times/costs. *European journal of operational research*, *246*(2), 345-378.

Beezão, A. C., Cordeau, J. F., Laporte, G., & Yanasse, H. H. 2017. Scheduling identical parallel machines with tooling constraints. *European journal of operational research, 257(3),* 834-844.

Burger, A. P., Jacobs, C. G., van Vuuren, J. H., & Visagie, S. E. 2015. Scheduling multi-colour print jobs with sequence-dependent setup times*. Journal of Scheduling, 18*, 131-145.

Calmels, D. 2019. The job sequencing and tool switching problem: state-of-the-art literature review, classification, and trends. *International Journal of Production Research, 57(15-16),* 5005-5025.

Calmels, D. 2022. An iterated local search procedure for the job sequencing and tool switching problem with non-identical parallel machines. *European Journal of Operational Research, 297(1)*, 66-85.

Cura, T. 2023. Hybridizing local searching with genetic algorithms for the job sequencing and tool switching problem with non-identical parallel machines. *Expert Systems with Applications, 223,* 119908.

Dang, Q. V., van Diessen, T., Martagan, T., & Adan, I. 2021. A matheuristic for parallel machine scheduling with tool replacements. *European Journal of Operational Research, 291(2),* 640 660.

da Silva, T. T., Chaves, A. A., & Yanasse, H. H. 2021. A new multicommodity flow model for the job sequencing and tool switching problem. *International Journal of Production Research, 59(12),* 3617-3632.

Franzin, A., & Stützle, T. 2019. Revisiting simulated annealing: A component-based analysis. *Computers & operations research*, *104*, 191-206.

Ghiani, G., Grieco, A., & Guerriero, E. 2007. An exact solution to the TLP problem in an NC machine. *Robotics and Computer-Integrated Manufacturing, 23(6),* 645-649.

Ghiani, G., Grieco, A., & Guerriero, E. 2010. Solving the job sequencing and tool switching problem as a nonlinear least cost hamiltonian cycle problem. *Networks: An International Journal, 55(4),* 379-385.

Laporte, G., González, J. J. S., & Semet, F. 2002, November. Exact algorithms for the job sequencing and tool switching problem. *In IV ALIO» EURO Workshop on Applied Combinatorial Optimization* (p. 71).

Liu, J., & Reeves, C. R. 2001. Constructive and composite heuristic solutions to the P//∑ Ci scheduling problem. *European Journal of Operational Research, 132(2),* 439-452.

Mara, S. T. W., Sutoyo, E., Norcahyo, R., & Rifai, A. P. 2023. The job sequencing and tool switching problem with sequence-dependent set-up time. *Journal of King Saud University Engineering Sciences, 35(1),* 53-61.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. 1953. Equation of state calculations by fast computing machines. *The journal of chemical physics, 21(6),* 1087-1092.

Oliveira, L. A. 2011. Dissertação e Tese em Ciências e Tecnologia segundo Bolonha. Lidel

Özpeynirci, S., Gökgür, B., & Hnich, B. 2016. Parallel machine scheduling with tool loading. *Applied Mathematical Modelling*, *40*(9-10), 5660-5671.

Paiva, G. S., & Carvalho, M. A. M. 2017. Improved heuristic algorithms for the job sequencing and tool switching problem. *Computers & Operations Research, 88,* 208-219.

Pezzella, F., Morganti, G., & Ciaschetti, G. 2008. A genetic algorithm for the flexible job-shop scheduling problem. *Computers & operations research*, *35*(10), 3202-3212.

Rifai, A. P., Mara, S. T. W., & Norcahyo, R. 2022-a. A two-stage heuristic for the sequence dependent job sequencing and tool switching problem. *Computers & Industrial Engineering, 163,* 107813.

Rifai, A. P., Sutoyo, E., Mara, S. T. W., & Dawal, S. Z. M. 2022-b. Multiobjective Sequence Dependent Job Sequencing and Tool Switching Problem. *IEEE Systems Journal, 17(1),* 1395 1406.

Solimanpur, M., & Rastgordani, R. 2012. Minimising tool switching and indexing times by ant colony optimisation in automatic machining centres. *International Journal of Operational Research*, *13*(4), 465-479.

Tang, C. S., & Denardo, E. V. 1988. Models arising from a flexible manufacturing machine, part I: minimization of the number of tool switches. *Operations research*, *36*(5), 767-777.

Anexo 1

ISEP INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

P.PORTO

DECLARAÇÃO DE INTEGRIDADE

DECLARAÇÃO DE INTEGRIDADE

Declaro ter conduzido este trabalho académico com integridade. Não plagiei ou apliquei qualquer forma de uso indevido de informações ou falsificação de resultados ao longo do processo que levou à sua elaboração.

Declaro que o trabalho apresentado neste documento é original e de minha autoria, não tendo sido utilizado anteriormente para nenhum outro fim.

Declaro ainda que tenho pleno conhecimento do Código de Conduta Ética do P.PORTO.

ISEP, Porto, 15 de setembro de 2024