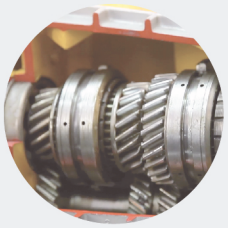
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA E GESTÃO INDUSTRIAL






Ferramenta de Apoio ao Escalonamento da Produção

PAULA CRISTINA DOS SANTOS OLIVEIRA REIS
novembro de 2020

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

FERRAMENTA DE APOIO AO ESCALONAMENTO DA PRODUÇÃO

Paula Cristina dos Santos Oliveira Reis

2020

Instituto Superior de Engenharia do Porto

Departamento de Engenharia Mecânica

P.PORTO


P.PORTO

Instituto Superior de Engenharia do Porto

FERRAMENTA DE APOIO AO ESCALONAMENTO DA PRODUÇÃO

Paula Cristina dos Santos Oliveira Reis

Estudante n.º 1160356

Dissertação apresentada ao Instituto Superior de Engenharia do Porto para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia e Gestão Industrial, realizada sob a orientação do Doutor André Borges Guimarães Serra e Santos e coorientação do Doutor João Augusto de Sousa Bastos.

2020

Instituto Superior de Engenharia do Porto

Departamento de Engenharia Mecânica

P.PORTO


P.PORTO

AGRADECIMENTOS

Um trabalho desta natureza para além do seu caráter singular, implicou a presença, participação e colaboração de um vasto conjunto de pessoas, sem as quais seria de todo impossível levar por diante a consecução, realização e alcance das metas propostas.

Ao Eng.º André Santos e ao Eng.º João Bastos pela orientação e coorientação, respetivamente, ao longo da execução deste projeto e pelo apoio e orientação científica prestado durante o desenvolvimento.

À minha família pelo apoio incondicional dado ao longo deste percurso.

Aos meus amigos pelo apoio, incentivo e motivação.

página propositadamente em branco

RESUMO

A imprevisibilidade e instabilidade dos mercados obrigam cada vez mais resposta rápida por parte das empresas. Todas as indústrias têm necessidade de investimento no sentido aprimorar as suas respostas, que deriva do crescimento do nível de competitividade dos mercados em que estão inseridas. A eficiência da sua produção é um dos fatores que mais condiciona a rentabilidade de uma organização.

Tendo isto em conta, há cada vez maior necessidade na eficácia do planeamento e programação da produção que permita maximizar os recursos instalados. O escalonamento da produção consiste na alocação e sequenciação das tarefas nas respetivas máquinas com o desígnio de encontrar a melhor sequência para o processamento das mesmas.

A presente dissertação tem por objetivo a criação de uma ferramenta de apoio à decisão, com intuito de colmatar as dificuldades no escalonamento da produção. Determinando uma afetação e sequenciação das tarefas às máquinas, minimizando uma determinada medida de desempenho.

Procedeu-se à descrição do problema de escalonamento, identificando formas de resolução e otimização dos diferentes tipos de problemas. Centrou-se os esforços na resolução de todos os tipos de problema de escalonamento, incluindo os problemas em *Job-Shop*, que consistem num conjunto de operações que têm de ser executadas numa máquina pré-determinada, obedecendo a um determinado sequenciamento com tempos pré-definidos. São caracterizados por uma complexidade *NP-hard* e sendo o protótipo capaz de resolver este tipo de problema fica automaticamente habilitado a resolver qualquer tipo de problema do escalonamento da produção.

A meta-heurística escolhida que permitiu a resolução dos problemas foi o *Simulated Annealing*, sendo escolhida por apresentar a vantagem de convergir para uma solução ótima, quando existe uma minuciosa escolha dos seus parâmetros combinado um arrefecimento muito lento. A finalidade desta meta-heurística foi construída para a minimização do *makespan*, ou seja, na minimização do tempo de fluxo total. O *framework* desenvolvido utiliza o SA, mas podem ser implementadas outras meta-heurísticas sem alterações significativas do modelo.

Como forma de demonstrar as potencialidades do *framework* foram analisadas instâncias de *Flow-Shop* e *Job-Shop*, quem têm por base conjuntos de problemas padronizados, previamente definidos por autores da área científica em questão.

Conclui-se a viabilidade da ferramenta criada e, em articulação com os procedimentos de otimização escolhidos, constituindo assim um fator diferenciador para as organizações e permitindo uma melhoria no desempenho dos recursos disponíveis.

PALAVRAS-CHAVE

Escalonamento, *Job-Shop*, *Flow-Shop*, *Makespan*, *Simulated Annealing*, Ferramenta de Apoio á Decisão, Otimização

página propositadamente em branco

ABSTRACT

The increasing unpredictability and instability of the markets requires a quick response from companies. All industries have a need for investment in order to improve their responses, which stems from the continuous growth in the level of competitiveness of the markets in which they operate. One of the factors that most affects the profitability of an organization is the efficiency of its production.

Bearing this in mind, there is an increasing need for effective production planning and scheduling to maximize the installed resources. The scheduling of production consists of the allocation and sequencing of tasks on the respective machines, with the aim of finding the best sequence for processing them.

This dissertation aims to create a decision support tool, in order to overcome the difficulties in scheduling production. Determining the assignment and sequencing of tasks to the machines, minimizing a certain measure of performance.

The scheduling problem was described, identifying ways to solve and optimize the different types of problems. Efforts were focused on solving all types of scheduling problems, including job shop problems, which consist of a set of operations that have to be performed on a pre-determined machine, following a specific sequence with pre-defined times. They are characterized by an NP-hard complexity and being the prototype capable of solving this type of problem, it is automatically enabled to solve any type of production scheduling problem.

The chosen meta-heuristic that allowed the resolution of the problems was Simulated Annealing, being chosen because it has the advantage of converging to an optimal solution, when there is a meticulous choice of parameters combined with very slow cooling. The purpose of this meta-heuristic was built to minimize the makespan, that is, to minimize the total flow time. The developed framework uses SA, but other meta-heuristics can be implemented without significant changes to the model.

As a way to demonstrate the framework's potential, instances of flow shop and job shop were analyzed, which are based on sets of standardized problems, previously defined by authors of the scientific area in question.

The viability of the created tool is concluded and, in articulation with the chosen optimization procedures, constituting a differentiating factor for the organizations and allowing an improvement in the performance of the available resources.

KEYWORDS

Scheduling, Job-Shop, Flow-Shop, Makespan, Decision Support Tool, Optimization

página propositadamente em branco

ÍNDICE

página propositadamente em branco

ÍNDICE DE FIGURAS

página propositadamente em branco

ÍNDICE DE TABELAS

página propositadamente em branco

LISTAS DE SIGLAS E SÍMBOLOS

Lista de Siglas

DTM	*Deterministic Turing Machine*
NDTM	*Non Deterministic Turing Machine*
SA	*Simulated Annealing*
OEE	*Overall Equipement Effectiveness*
AE	Algoritmos Evolucionários
PSO	*Particle Swarm Optimization*
WIP	*Work In Progress*

página propositadamente em branco

1. INTRODUÇÃO

Neste capítulo são apresentados, de forma sucinta, os aspetos que motivaram a elaboração desta dissertação assim como os objetivos que a orientaram, sendo exposta uma pequena explicação do contexto em que se insere o trabalho abordado. Relatam-se também as contribuições e resultados que se pretende alcançar e o modo como se quer atingir os objetivos traçados. O capítulo termina com a apresentação da estrutura do trabalho e com a descrição do propósito de cada capítulo.

O objetivo principal da introdução é apresentar o trabalho, preparando e orientando a leitura dos capítulos seguintes. Para tal, serão abordados vários aspetos pertinentes como o contexto e o enquadramento, passando pela pertinência, importância e atualidade da questão central deste trabalho, os objetivos também serão expostos, bem como, as questões que os orientaram. A inclusão das opções metodológicas e do plano de investigação tem como finalidade, apenas, de demonstrar quais as opções tomadas para satisfazer os objetivos traçados, por isso não serão descritas de forma exaustiva. No final, deste capítulo será exposta a estrutura do trabalho, descrevendo sucintamente o propósito de cada capítulo.

1.1. Enquadramento e pertinência

A evolução das condições de competitividade económica justifica a mudança de paradigmas. As empresas são pressionadas para atingir grandes níveis de qualidade, a um custo cada vez menor e num prazo diminuto. A globalização dos diferentes tipos de mercados, impulsiona o investimento das organizações de forma a melhorar e otimizar todos os processos que compreendem as suas atividades.

As empresas, devem dotar-se de métodos e ferramentas para a organização e controlo da produção. Torna-se cada vez mais pertinente a criação de ferramentas para rentabilizar os equipamentos e melhorar os procedimentos de forma a obter taxas de eficácia cada vez mais elevadas. Oferecer soluções de escalonamento eficientes e rentáveis que permitam resolver problemas complexos como o aumento da eficiência das máquinas e diminuir os desperdícios. Surge assim, a necessidade de desenvolvimento das ferramentas, com a finalidade de permitir resolver problemas específicos mesmo que isso não implique encontrar a melhor solução no tempo útil disponível.

O problema da programação de produção tem sido objeto de estudo, pois a eficiência e a taxa de utilização dos recursos das empresas dependem diretamente da alocação dos recursos. Ou seja, tudo depende diretamente do uso pleno dos meios e a minimização dos custos (Wang, Ping; Sang, Hongyan; Guo, Hengwei; Li, 2020).

O objetivo do escalonamento da produção é alocar com eficiência nas máquinas disponíveis os trabalhos ou operações e o seu subsequente faseamento temporal. Como medidas de resolução tem-se implementado metodologias de simulação, modelos analíticos e heurísticas, no entanto com a tendência de crescimento da complexidade dos sistemas de fabricação tem sido cada vez maior a necessidade de investigação deste problema (Shahzad, Atif; Mebarki, 2016).

Muitos dos problemas do planeamento da produção envolvem a otimização de um conjunto de objetivos. O problema está relacionado com a atribuição de *n* trabalhos a *m* máquinas, com a

finalidade de minimizar as funções objetivo em questão. Sendo que são considerados vários objetivos que se pretendem atingir, minimizando ou maximizando os mesmos e há, também, uma maior variedade de soluções para esses problemas. Um dos problemas mais conhecidos nesta situação, são os problemas em *Job-Shop*, caracterizados por tarefas que têm de passar por um determinado número de máquinas, seguindo uma sequência pré-definida com a finalidade de determinar a prosseguimento em cada máquina satisfazendo as regras especificas. Devido à sua alta complexidade é muito árduo encontrar uma solução ideal num intervalo de tempo razoável, pois tendo uma rota de trabalho fixa e a cada operação de um trabalho é alocada uma máquina exclusiva para o processamento leva a um efeito gargalo das máquinas mais utilizadas no *shop floor* (Binh Ho & Cing Tay, 2018).

Considerando o grande número de entidades integradas e as suas interações no sistema produtivo, a complexidade que daí advém é uma das causas dos problemas de escalonamento. Isto prende-se que estas entidades precisam de ser geridas de acordo com um determinado cronograma e com as restrições inerentes ás tarefas. Dessas restrições pode-se mencionar as de precedência, o transporte, data de conclusão e disponibilidade de recursos(Nouiri, Bekrar, Jemai, Niar, & Ammari, 2018). A programação das operações envolve a alocação dos recursos durante um período de tempo para executar um conjunto de tarefas. Obter as melhores soluções para os problemas de escalonamento é de grande importância para a indústria, uma vez que a taxa de produção e as despesas de qualquer planta de produção dependem dos cronogramas usados. Uma programação assertiva permite que a empresa utilize os seus recursos de maneira eficaz e atinja os objetivos estratégicos (Motaghedi-larijani, Sabri-laghaie, & Heydari, 2010).

O escalonamento da produção, tem como objetivo determinar a sequência mais adequada para o processamento de um conjunto de encomendas ou tarefas, de modo a otimizar os critérios considerados de interesse para a organização. Assim, o problema de escalonamento compreende dois subproblemas. Numa primeira fase trata de dar resposta a questões de afetação, determinando que recursos deverão ser alocados a que tarefas, de modo a otimizar e alcançar os objetivos da organização. Posteriormente, pretende responder a questões de calendarização/ sequenciação, devendo para tal definir a distribuição no tempo as tarefas (Pinedo, 2008).

Existem vários tipos de problemas de escalonamento para um sistema de produção que diferem de acordo com a definição do seu intervalo e organização. Um dos problemas mais difíceis são os problemas em *Job-Shop*, em que um conjunto de tarefas deve ser processado por um conjunto de máquinas. Cada trabalho é formado por uma sequência de operações consecutivas e cada operação requer exatamente uma máquina. A decisão diz respeito à sequência das operações nas máquinas, onde um determinado indicador de desempenho deve ser otimizado (Pezzella, Morganti, & Ciaschetti, 2008). O propósito é encontrar uma alocação para cada operação e definir a sequência de operações em cada máquina para minimizar o tempo máximo do *makespan*, isto é, minimizar o tempo em que a última tarefa é realizada (Nouiri et al., 2018).

Pela natureza NP dos problemas em *Job-Shop*, muitos métodos eficientes de heurísticas e meta-heurísticas são desenvolvidos para obter soluções quase ótimas que satisfazem as restrições e minimizam ou maximizam a função objetivo. Entre estes métodos temos o *Simulated Annealing (SA)* que será abordado como um algoritmo eficaz na otimização de um problema deste tipo (Pérez & Raupp, 2016).

Em suma, o desenvolvimento de metodologias cada vez mais sustentáveis, tanto a nível de qualidade como de custo por parte das organizações torna-se vital. Daí a pertinência da criação de uma ferramenta genérica para a resolução de problemas de escalonamento específicos envolvendo meta-heurísticas.

1.2. Questão e objetivos de investigação

Com o desenvolvimento deste trabalho pretende-se colmatar as necessidades da gestão e do planeamento da produção em reduzir cada vez mais os desperdícios inerentes a um processo de produção. Assim, o objetivo desta dissertação centra-se na criação de uma ferramenta de apoio à decisão com o intuito de agilizar o escalonamento da produção.

Os objetivos específicos que norteia o presente trabalho, é o desenvolvimento de uma ferramenta de apoio ao escalonamento da produção que, permita ao utilizador a resolução de forma genérica de problemas específicos de escalonamento. Pretende-se, que a resolução do problema seja conseguida através de meta heurísticas, culminando no final com uma comparação de resultados. Como método de validação da ferramenta, serão executados problemas com características adequados.

Assim os objetivos específicos são caracterizados por:

1. Desenvolvimento de um modelo de resolução de problemas de escalonamento
2. Desenvolvimento de um protótipo da rede de fluxo das tarefas
3. Validação e análise dos resultados da solução de problemas com características adequados.

1.3. Opções metodológicas

Neste capítulo o intuito é apresentar as opções metodológicas levadas em conta para a realização do trabalho proposto. O método que esclarece o procedimento lógico usado é o método hipotético-dedutivo, pois o conceito é designado por combinar o método indutivo com o método dedutivo, ou seja, reformula-se a teoria com base na experiência e com base na teoria capta-se a experiência e a realidade (Gil, 2008). Isto significa que, a base da ferramenta de escalonamento da produção que será desenvolvida sobre teoria, ou seja, os métodos aproximados de resolução já conhecidos serão aplicados e programados na ferramenta para depois serem testados através de problemas adequados em cada caso.

De encontro ao referido, a abordagem de investigação passará por ser quantitativa, isto é, vai-se proceder ao teste das teorias objetivas, testando os valores obtidos para as medidas de desempenho mais apropriadas aos problemas selecionados e a posterior análise estatística.

As técnicas de recolha de dados incidem sobre a análise de documentos, no qual se realizou uma recolha, leitura e análise de declarações em teses e em artigos científicos da área, conjuntamente com as experiências, ou seja, em que as hipóteses formuladas tiveram em conta a correlação das variáveis (Macedo, Zacarias, & Tribolet, 2002).

Por último, o plano de investigação, centrar-se-á na simulação, isto é, procede-se a criação de vários cenários para o estudo da minimização da função objetivo definida na programação da produção, com o intuito de simular o comportamento do sistema (Macedo et al., 2002).

1.4. Estrutura do trabalho

O presente documento é composto por 5 capítulos principais.

Primeiramente e já apresentado, o capítulo da Introdução, onde foram definidos o enquadramento e pertinência do problema em estudo, as questões e objetivos de investigação do mesmo e descrevendo sucintamente as opções metodológicas. Seguidamente será abordado a Revisão Bibliográfica.

Na revisão bibliográfica realizou-se o enquadramento científico e técnico dos conteúdos integrantes desta dissertação. É dirigida nas temáticas de escalonamento da produção com foco na resolução de problemas de alocação e sequenciamento. São abordados os diferentes tipos de implementação industrial, nomeadamente os problemas em *Job-Shop* e *Flow-Shop*. Consecutivamente as medidas de desempenho que permitem avaliar o desempenho dos métodos de resolução implementados. Com isto, e após a abordagem á teoria da complexidade são caracterizados os métodos de otimização existentes e posteriormente especifica-se mais as meta-heurísticas.

No terceiro capítulo, denominado por descrição do problema e implementação computacional clarifica-se o problema inerente ao objetivo que se pretende alcançar, assim como a apresentação do método de resolução capaz de automatizar e otimizar a programação da produção.

O penúltimo capítulo exibe as experiências computacionais realizadas, analisando e comparando os resultados obtidos para as instâncias de teste reconhecidas na literatura, com um grau de complexidade elevado, com a finalidade de validar a ferramenta de apoio á decisão criada.

Por forma a concluir o trabalho desenvolvido ao longo desta dissertação, são expostas as conclusões que foram possíveis apurar, as limitações do mesmo e descrevem-se algumas perspetivas de desenvolvimento futuro.

2. REVISÃO BIBLIOGRÁFICA

Este capítulo é dedicado ao enquadramento científico e técnico dos conteúdos integrantes desta dissertação. A revisão bibliográfica é dirigida às temáticas de escalonamento da produção com foco na resolução de problemas de alocação e sequenciamento. Com a representação dos problemas de escalonamento, advém a implementação industrial, pois os problemas de escalonamento são tratados de igual forma em ambientes com os mesmos padrões e características e poderão ser divididos em problemas de máquina única, máquina paralela, linhas de fabrico, oficinas de fabrico e *Open Shop*.

De modo a avaliar e computar as soluções de escalonamento, desenvolveu-se a secção das medidas de desempenho, sejam de produtividade ou de pontualidade. Seguidamente, e indo de encontro aos problemas e dificuldades existentes na implementação do escalonamento de produção, aborda-se as restrições tecnológicas, a teoria de complexidade e tipos de problemas de otimização.

A exposição é desenvolvida com base em pesquisas bibliográficas, as quais permitiram recolher e reunir diferentes contribuições nos métodos de otimização exatos e aproximados, a forma como é possível representar, codificar os problemas e a avaliação dos mesmos. Finalizando este capítulo, sintetiza-se algumas meta-heurísticas existentes e apresenta-se alguns comentários finais aos conteúdos expostos. Este levantamento bibliográfico, permitiu assim, caracterizar a pertinência científica das temáticas envolvidas no projeto e enquadrar o seu contexto.

2.1. Escalonamento da Produção

Em mercados cada vez mais competitivos, é fundamental que as organizações façam um escalonamento das operações eficaz. O escalonamento da produção é considerado um problema de decisão com o qual se deparam regularmente muitas indústrias.

O escalonamento da produção define-se pela organização ao longo do tempo da execução de um conjunto de tarefas, levando em consideração restrições de tempo, tais como prazos, restrições de precedência e restrições de capacidade dos recursos necessários para executar as tarefas vigentes. Descreve, a execução das tarefas e alocação dos recursos ao longo do tempo, com o propósito anteriormente delineado. É importante perceber que subsistem dois tipos de problemas de planeamento, o problema de alocação caracterizado pela decisão de uma data de início e uma data de termo para cada tarefa e o oposto, problema de sequenciamento, especializado nas tarefas descritas por relações de precedência que competem pelo uso do mesmo recurso (Jacek Blazewicz, Ecker, Pesch, Schmidt, & Weglarz, 2007)

As estratégias de escalonamento, são definidas como estáticas ou dinâmicas. No escalonamento estático, os trabalhos são conhecidos no início da programação e o sequenciamento é mantido fixo durante todo o processo de fabricação. Por outro lado, no escalonamento dinâmico pode-se incorporar novos trabalhos, as datas de início e fim podem ser alteradas assim como o estado de um conjunto de máquinas. Todas estas alterações que podem ser incorporadas, levam a uma necessidade de reagendamento que pode ser periódica, caracterizada por mudanças em intervalos regulares e orientada a eventos causada pela introdução de um novo trabalho, ou, avaria de máquinas (Koskinen, Raduly-Baka, Johnsson, & Nevalainen, 2020).

No processo de programação da produção, precisa-se de saber o tipo e quantidade de cada recurso, para que seja possível determinar quando as tarefas podem ser executadas de maneira viável. Ao especificar os recursos, define-se o limite do problema de escalonamento e além disso descreve-se cada tarefa como a sua duração, as datas de iniciação e conclusão. Dever-se-ia ainda descrever quais queres restrições tecnológicas, isto é, restrições de precedência existentes entre tarefas. As informações sobre os recursos e tarefas definem o problema, mas encontrar a solução é um assunto bastante complexo (Pinedo, 2008).

Uma programação da produção constitui uma solução para o problema de escalonamento. Descreve a execução de tarefas e a alocação de recursos ao longo do tempo, com o intuito de atingir um ou mais objetivos. O escalonamento induz necessariamente, um conjunto de relações de sequenciamento, descrita em termos de relações de precedência.

O escalonamento da produção é caracterizado pela priorização das atividades, que devem ser realizadas em uma dada sequência para atender a um ao mais objetivos. Os diversos problemas de escalonamento têm sido extensivamente estudados nas últimas décadas em diferentes áreas com respeito à modelagem e algoritmos. Isto porque o problema de organizar a execução de um conjunto de tarefas por um grupo finito de recursos, encontra-se presente em quase todo o tipo de atividade produtiva e pode ser descrito, embora constituía um impasse devido a sua complexidade matemática (Sule, 1997).

Os objetivos da programação da produção são aumentar a utilização de recursos, reduzindo o stock em processo e o atraso na entrega dos trabalhos (Martins & Sacomano, 1994). Os critérios de otimização do processo de escalonamento, que as organizações consideram mais relevante é o cumprimento das datas de entrega, seguido pela maximização da utilização das máquinas do sistema, a minimização dos *work in progress* (*WIP*) e a minimização das mudanças de ferramenta, culminando tudo na maximização da produtividade (Smith, 1986).

O escalonamento da produção é afetado pelo planeamento, programação, controlo da produção, que lida com o plano de médio e longo prazo de toda a organização. Tem como função otimizar o *mix* de artigos da organização a longo prazo, com base nos níveis de stock, previsões da procura e requisitos dos recursos.

Considera-se que o escalonamento da produção é constituído pela alocação ou atribuição das operações das tarefas aos recursos do sistema e o respetivo sequenciamento ou calendarização. Estas fases podem ser tratadas independentemente ou integradas, dependendo do tipo e dimensão do sistema em questão e da estratégia de escalonamento adotada.

2.1.1. Problema de Alocação

Os problemas de alocação correspondem à afetação das tarefas a uma dado conjunto de máquinas, em determinados instantes do tempo (Pinedo, 2008).

Blazewicz (2007) descreveu o problema de alocação como n tarefas pertencentes a $T = \{T1, T2, \ldots., Tn]$, *m* máquinas pertencentes a $P = \{P1, P2, \ldots, Pm\}$ e *s* recursos adicionais pertencentes a *R={R1, R2,…, Rs},* em que a questão se centra na forma que deverão as máquinas *P* e os recursos *R*, ser afetados às tarefas de *T* por forma a maximizar/minimizar uma determinada função objetivo. As restrições a que estão, normalmente, sujeitas este tipo de problema são uma tarefa apenas poder ser executada por uma máquina de cada vez e que cada máquina apenas poder

processar uma tarefa de cada vez, sendo importante a impossibilidade de interromper o processamento de uma tarefa que esteja a recorrer.

2.1.2. Problema de Sequenciamento

Um problema de sequenciamento da produção é a decisão a ser tomada sobre a ordem em que as tarefas serão executadas, sendo que a programação da produção envolve a consideração de uma série de trabalhos que disputam vários recursos por um período, recursos esses que possuem capacidade limitada (Pacheco & Santoro, 1999).

O objetivo fundamental de um problema de sequenciamento da produção compreende afetar os recursos de produção as tarefas ou operações, que deverão ser realizadas, segundo uma determinada ordem, ou o procedimento contrário, isto é, atribuir as tarefas aos recursos. Assim, o sequenciamento pode ser entendido como a atribuição de recursos no tempo para o processamento de um conjunto de tarefas (Baker, 1974).

O objetivo é encontrar uma prossecução das tarefas atribuídas aos recursos, correspondendo a um plano exequível e ótimo em relação a um qualquer critério de otimização adotado (French, 1982). Segundo Portman (1997) o sequenciamento é visto como a definição de tempos de início e de conclusão, e a atribuição dos recursos a cada tarefa de um dado conjunto, obedecendo às várias restrições inerentes.

2.2. Representação de Problemas de Escalonamento

A forma de resolução de um problema de escalonamento, depende das características do mesmo e torna-se rentável recorrer a uma forma de nomenclatura para os classificar e facilitar a sua especificação e representação.

Um dos métodos de classificação encontrados é a notação de quatro campos: $A|B|C|D$ descrito por Conway (1967) , sendo que *A* descreve o número de trabalhos (em problemas estáticos) ou a filosofia de chegada (em problemas dinâmicos), *B* é definido pelo número de recursos, *C* o tipo de sistema e *D* o critério de otimização.

A classificação de French (1982) tem os parâmetros $n|m|A|B$, em que *n* é o número de parâmetros, podendo este ser um valor inteiro, ou mesmo *n*, caso não esteja definido, o parâmetro *m* é o número de processadores no sistema e *A* o sistema de produção, sendo que quando *m-1*, A é nulo. Para finalizar, o critério de otimização é dado pelo parâmetro *B*.

Por último, apresenta-se o esquema de classificação de Graham, que se torna o mais apropriado quando estamos perante problemas que envolvam a possibilidade de interrupção, precedências entre tarefas, existência de tempos de preparação, datas de entrega efetivas, e datas de lançamento. A nomenclatura é dada por $\alpha|\beta|\gamma$, em que α simboliza o tipo de problema, isto é, problema em máquina única, máquinas paralelas, *Open-Shop, Flow-Shop* ou *Job-Shop,* β indica as características da tarefa, referente as restrições de precedência, se β for nulo assume-se que as tarefas estão disponíveis para processamento ao mesmo tempo, e por último γ que identifica o critério de otimização, isto é, a função objetivo (Coffman & R. L. Graham, 1971; Graham, Lawler, Lenstra, & Kan, 1979).

Contrariamente, a todas as nomenclaturas apresentadas anteriormente, há autores que preferem utilizar uma linguagem natural. Esta tem a vantagem de ser mais intuitiva, no entanto é uma classificação mais subjetiva, o que muitas vezes não vai de encontro ao pretendido, uma classificação objetiva, clara e estruturada (Baker, 1974)(Morton & Pentico, 1993).

2.3. Implementação Industrial

Os problemas de escalonamento são, geralmente, classificados como (Pinedo, 2008):

1. Problemas em máquina única
2. Problemas em máquinas paralelas
3. Problemas em *Job-Shop*
4. Problemas em *Flow-Shop*
5. Problemas em *Open-Shop*

De seguida iremos abordar os diferentes tipos de problemas de escalonamento.

2.3.1. Problema em Máquina Única

A programação da produção em máquina única, é caraterizada por um problema de sequenciamento puro, no qual existe um único recurso, ou máquina. Este tipo de problema ilustra uma variedade de tópicos de escalonamento num modelo tratável, fornece um contexto para investigar muitas medidas de desempenho diferentes e várias técnicas para encontrar a solução. Muitas vezes, um problema de máquina única aparece embutido num problema complexo, que pode ser resolvido independentemente e posteriormente ser incorporado, acontece regularmente em processos que apresentam *bottleneck* (Baker & Trietsch, 2009).

O problema básico é caracterizado pelas seguintes condições:

1. há *n* trabalhos de operação única disponíveis simultaneamente para processamento (em tempo zero);

2. as máquinas podem processar no máximo um trabalho de cada vez;

3. os tempos de configuração dos trabalhos são independentes da sequência do trabalho e estão incluídos nos tempos de processamento;

4. a máquina está disponível constantemente;

5. a máquina nunca é mantida desocupada enquanto o trabalho está em espera;

6. Quando uma operação começa ela continua sem interrupção (Baker & Trietsch, 2009).

Nestas condições, há uma correspondência individual entre uma sequência dos *n* trabalhos e uma permutação dos índices de trabalho 1,2, … *n*. Portanto o número total de soluções possíveis para um problema de máquina é *n!*. Um caso clássico dos problemas de máquina única é o escalonamento de *n* tarefas em que o objetivo é reduzir o atraso máximo do sistema. Cada tarefa *j* tem um tempo de processamento *pj* associado, assim como uma data de entrega *dj* , em que para qualquer sequência assumida para a execução das tarefas tem um instante de conclusão *Cj* e um

atraso associado *Lj=Cj-dj*, sendo que não é possível garantir a entrega de todas as tarefas antes, ou no limite, do prazo correspondente de entrega (Madureira, 2003).

Em geral, o planeamento requer decisões de sequenciamento e afetação de recursos, mas neste caso, em que há apenas um recurso, a alocação desse recurso é completamente determinada por decisões de sequenciamento, ou seja, não existe distinção entre sequenciamento e alocação de recursos (Baker & Trietsch, 2009).

2.3.2. Problemas em Máquinas Paralelas

O planeamento de problemas em máquinas paralelas, possibilita a realização de um conjunto de tarefas em simultâneo, em que cada trabalho é constituído apenas por uma operação. Funcionam como um conjunto de problemas de máquinas únicas, integrado num sistema global que se realizam em paralelo, com a mesma função (Jacek Blazewicz, Domschke, & Pesch, 1996). O modelo básico, caracterizado por *n* trabalhos disponíveis simultaneamente no momento zero para serem processados em *m* máquinas paralelas disponíveis, assumindo que um trabalho pode ser processado no máximo uma vez por máquina (Baker & Trietsch, 2009).

É um problema encontrado frequentemente em aplicações reais, determinado pelo facto de que várias máquinas podem ser usadas para a execução de um trabalho, que requer apenas uma delas. A resolução passa pelo processo de decisão de alocação e sequenciamento, geralmente é executada em duas etapas (Lopez & Roubellat, 2008):

1. decisão em qual máquina cada operação será executada;

2. determinação da sequência de operações em cada máquina;

Cada ordem de trabalho, tem estabelecida datas de entrega e as suas restrições, que obrigam a um escalonamento ponderado das tarefas necessárias à realização do produto final, que conjuntamente com o lançamento de várias ordens de produção em simultâneo, o tempo necessário para a realização de cada tarefa, as precedências obrigatórias tornam o processo de escalonamento complexo. Este tipo de problema é geralmente dividido em três classes dependendo se as máquinas são (Lopez & Roubellat, 2008):

1. Idênticas: o tempo de processamento é o mesmo para todas as máquinas;
2. Uniformes: o tempo de processamento varia de acordo com o desempenho da máquina;
3. Independentes: o tempo de processamento é completamente variável entre as diferentes máquinas.

2.3.3. Problemas em *Flow-Shop*

Em *Flow-Shop* um trabalho é divido em operações, e cada operação é executada em uma máquina diferente. Assim sendo um trabalho é uma coleção de operações com uma estrutura de precedências, ou seja, cada trabalho exige que uma sequência específica de operações. As condições que caracterizam os modelos de *Flow-Shop*, são similares com os problemas de máquina única definidos por (Baker & Trietsch, 2009):

1. um conjunto de *n* trabalhos não relacionados e de várias operações, está disponível para processamento no tempo zero (t=0). Cada trabalho, requer *m* operações e cada operação requer uma máquina diferente);

2. os tempos de configuração para as operações são independentes da sequência e compreendidos nos tempos de processamento;

3. a descrição das tarefas é conhecida antecipadamente;

4. todas as máquinas estão sempre disponíveis;

5. quando uma operação começa, ela continua sem interrupção.

Os problemas de linhas de fabrico são definidos por cada trabalho consistir em m operações *Oij* com tempos de processamento *pij* (*j=1,…,m*) em que *Oij* deve ser processado na máquina *Mj* e existem restrições de precedência no formato *Oij* → *Oi,j+1* para cada *i = 1,…, n*, ou seja, cada trabalho é processado primeiro na máquina 1, depois na máquina 2 e assim sucessivamente. Com isto surge a problemática de encontrar uma ordem de serviço πj para cada máquina *j* (Brucker, 2006).

2.3.4. Problemas em *Job-Shop*

Os problemas clássicos de *Job-Shop* diferem dos de F*low-Shop* na questão de que o fluxo de trabalho não é unidirecional. Consiste, num ambiente de produção onde um conjunto de máquinas diferentes processa as operações das tarefas. Cada uma das tarefas que é executada, tem uma sequência específica de processamento para percorrer as máquinas, ou seja, cada tarefa é composta por um conjunto de operações, sequenciado, que se caracteriza pela máquina onde está a ser realizada e pelo tempo de processamento associado (Pinedo, 2008).

Os sistemas de fabricação *Job-Shop* são caracterizados por cada ordem de fabrico possuir um conjunto de operações que obedece uma ordem especifica, na qual *n* ordens de fabrico vão ser processadas em m máquinas (Baker & Trietsch, 2009). A resolução do problema *Job-Shop* tem como finalidade principal a obtenção de uma sequência segundo o qual as operações deverão ser realizadas, tendo em vista a concretização de determinados critérios de desempenho.

Problemas de *Job-Shop* são qualificados como *NP-hard*, significando que um algoritmo ótimo para o resolver requer um número de etapas que cresce exponencialmente com a dimensão dos dados do problema. Como tal, estão normalmente associadas duas dificuldades, a primeira relacionada com a complexidade exponencial que no limite as soluções possíveis e exequíveis são dadas por $(n!)^m$. A segunda dificuldade diz respeito com a diversidade de restrições que o problema pode estra sujeito, tais como, a existência de datas de lançamento e de entrega de tarefas, a disponibilidade dos recursos, as restrições de precedência entre as tarefas entre outras (Pinedo, 2008).

Existe ainda o conceito de J*ob-Shop* flexível como sendo uma extensão do problema clássico, no qual uma operação pode ser processada em máquinas alternativas, isto é, pelo menos uma operação tem disponível um conjunto de máquinas alternativas, cujos tempos de processamento podem variar com a máquina (Brandimarte, 1993).

2.3.5. Problemas em *Open-Shop*

Se a restrição da ordenação obrigatória das operações das tarefas segundo a sua precedência num problema *Job-Shop*, for relaxada ou eliminada, estamos perante um problema do tipo O*pen-Shop* (Fang, Ross, & Corne, 1993). A ausência de uma disposição imposta entre as operações, torna este tipo de problemas combinatórios e impede a aplicação de técnicas de resolução já comprovadas em ambientes de F*low-Shop* e J*ob-Shop*.

Os problemas em O*pen-Shop*, são dados por um conjunto *T* de *n* trabalhos *T1, T2, …, Tn* que devem ser executados sobre um conjunto *M* de *m* máquinas especializadas. Considera-se que, se cada tarefa Ti contiver m operações *(i,j*), *J=1,2,…, m* a operação *(i,j)* ocorrerá na máquina *mj* (Lopez & Roubellat, 2008).

Consequentemente, o ponto que define os problemas de *Open-Shop* é a inexistência de relações de precedência entre as operações, ou seja, o problema centra-se então em encontrar ordens de serviço, ordens de operações pertencentes ao mesmo trabalho, e ordens da máquina, que dizem respeito as ordens de operação a serem processadas na mesma máquina (Brucker, 2006).

Comparando os problemas de F*low-Shop*, J*ob-Shop* com os problemas de O*pen-Shop*, denota-se que este é um problema caracterizado *NP-hard*, sem restrições respetivas à rota de processamento das tarefas e da alocação das tarefas nas máquinas, a dificuldade reside no crescimento impetuoso do espaço de soluções à medida que o tamanho do problema aumenta (Zhuang et al. , 2019).

2.4. Medidas de Desempenho

De modo a ser aplicado o melhor método, que leve a um desempenho pretendido do sistema de produção, devem ser considerados certos critérios e medidas de desempenho para justificar a seleção do método mais adequado à resolução do problema intrínseco. As medidas de desempenho permitem avaliar a eficiência do programa de escalonamento, de encontro a um objetivo final, que poderá ser de maximização ou minimização dos critérios referidos (Varela, 2007).

As medidas de desempenho, pretendem otimizar fatores internos e/ou externos, podendo estar relacionados com a produtividade, isto é, cujo foco principal é a eficiência do escalonamento e com a pontualidade, alusivas ao cumprimento das datas de entrega acordadas.

Segundo Varela (2007), os critérios de otimização permitem atingir diversos objetivos, nomeadamente:

1. maximização do fluxo de produção;

2. satisfação dos requisitos de qualidade e rapidez da resposta aos clientes;

3. minimização dos custos de produção;

4. combinação dos casos anteriores;

As medidas mais usuais na função objetivo de um problema de escalonamento são apresentadas de seguida (Brucker, 2006; Pinedo, 2008; Tavares, 2015):

Minimização do *Makespan*

O *makespan* é simbolizado por $C_{max,}$ é definido pelo tempo em que a última tarefa de uma sequência de trabalhos está completa, onde *Cj* representa a data de conclusão da tarefa *j*. Para tarefas constituídas por mais de uma operação, representa o momento em que a última é executada (Blazewicz et al. , 2004).

$$C_{max} = max\ (C_j) \quad (1)$$

Minimização do tempo de fluxo total

O tempo de fluxo total é igual à soma dos tempos de conclusão *Cj* de todas as tarefas. A função objetivo é dada por:

$$min \sum_{j=1}^{n} C_j \quad (2)$$

Minimização do atraso máximo

O atraso máximo, designado por T_{max}, diz respeito à tarefa com maior diferença *Tj* entre o instante de conclusão *Cj* e a data de entrega *dj*, quando a tarefa é terminada após a data de entrega, assim temos:

$$Tj = max\{0; Cj - dj\} \quad (3)$$

É de notar que *Tj* não poderá tomar valores inferiores a zero, pois isso significaria um adiantamento em vez de um atraso. A função objetivo resultante é:

$$min\ Tmax \quad (4)$$

Minimização da soma dos atrasos

A função objetivo consiste em determinar a menor soma de todos os atrasos Tj, isto é:

$$min \sum_{j=1}^{n} Tj \quad (5)$$

Minimização do número de tarefas atrasadas

O objetivo é, como o nome indica, minimizar o número de tarefas atrasadas e pode ser formulado da seguinte forma:

$$min \sum_{j=1}^{n} y_j \quad (6)$$

Onde,

$$yj = \begin{cases} 1, & se\ a\ tarefa\ j\ está\ atrasada \\ 0, & caso\ contrário \end{cases} \quad (7)$$

Minimização do atraso e antecipação máximo

O atraso e antecipação, *lateness*, dado por *Lj* é a diferença entre a data de conclusão de uma tarefa e o prazo de conclusão. Se for positivo indica um atraso na entrega e se for negativo significa uma antecipação ao prazo de entrega.

Sendo L_{max}= Max(Lj), o *lateness* máximo, a formulação do problema é apresentada da seguinte forma:

$$min\, Lmax, em\ que, Lj = Cj - dj \qquad 8$$

Minimização do número de tarefas em atraso

Diz respeito a problemas de minimização do número de tarefas em atraso, ou seja, a função objetivo pretende minimizar o número de tarefas que não cumprem a data de entrega:

$$min \sum_{j=1}^{n} Uj \qquad 9$$

2.5. Teoria da Complexidade

Os problemas de escalonamento são muito complexos e de resolução computacionalmente difícil devido a sua diversidade, à sua natureza dinâmica e a sua dimensão. Normalmente são caracterizados por serem problemas de otimização, ou seja, problemas em que o objetivo é a minimização ou maximização de um ou mais critérios de otimização. Por norma os problemas de escalonamento são primeiramente formulados através dos problemas de decisão, com o intuito de se obter respostas binárias de alocação de trabalhos numa determinada máquina do sistema (Varela, 2007).

O escalonamento da produção abarca dois tipos de problemas: problemas de decisão e problemas de otimização. Os problemas de decisão apresentam respostas binárias, enquanto que os problemas de otimização têm respostas mais complexas que correspondem a uma solução que otimiza uma medida de desempenho.

Existem dois principais tipos de algoritmos no ponto de vista da teoria da complexidade, os algoritmos de tempo polinomial e exponencial. Um algoritmo de tempo polinomial é aquele cuja função de complexidade temporal é *O(p(k)),* em que p é um polinómio e k é o comprimento de entrada de uma instância. Qualquer que seja o algoritmo cujo a função de complexidade de tempo não pode ser delimitada desta maneira, diz-se que é um algoritmo de tempo exponencial (Jacek Blazewicz et al., 2007).

É possível fazer uma distinção entre classes de problemas. Existe a classe de problemas P, que consiste em todos os problemas de decisão que poderão ser resolvidos através da máquina determinística de *Turing* (*DTM- Deterministic Turing Machine*), ou seja, um modelo abstrato de computação, em tempo superiormente limitado por um polinómio do comprimento de entrada. E a classe de problemas NP, caracterizada também pelos problemas de decisão mas que são resolvidos em tempo polinomial por uma máquina não determinística de *Turing* (*NDTM- Non Deterministic Turing Machine),* esta classe ainda pode ser subdivida em *NP-hard* e no tipo *NP-complete* (Jacek Blazewicz et al., 2007). Os problemas de otimização têm respostas mais complexas e dizem respeito a uma solução que otimiza um critério de desempenho, na sua maioria,

enquadram-se na classe de problemas NP. Para a resolução dos problemas de otimização NP não existem algoritmos eficientes e por este motivo recorre-se a métodos de aproximação que conseguem fornecer uma resposta suficientemente boa, mas não ótima(Papadimitrou & Kenneth, 1998).

Devido à natureza intrinsecamente dinâmica dos problemas reais de escalonamento, estes são de resolução complexa, para os quais o tempo requerido para a determinação de um plano ótimo ou quase ótimo aumenta exponencialmente com o tamanho do problema (Pentico & Morton, 1993).

2.6. Métodos de Otimização

Os métodos para a resolução de problemas de otimização podem ser classificados como métodos exatos ou aproximados. O objetivo dos métodos exatos é encontrar a solução ótima para o problema recorrendo a uma pesquisa exaustiva do espaço de soluções, sendo, habitualmente, apenas utilizados em problemas de pequena dimensão. Contrariamente, os métodos de aproximação têm o objetivo de encontrar soluções satisfatórias em tempo útil. A utilização dos métodos aproximados não garante o ótimo global, mas sim soluções muito próximas dessa. São aplicados a problemas de grande dimensão, uma vez que são eficientes em termos de tempo (Pinedo, 2008).

2.6.1. Métodos Exatos

Os métodos exatos de resolução de problemas de escalonamento compreendem o método de *Branch and Bound*, programação dinâmica e programação por restrições (El-Ghazali, 2009). Este tipo de métodos considera todo o espaço de pesquisa, e o problema é resolvido subdividindo-se em problemas mais simples.

Estes métodos exploram as soluções possíveis para um problema no sentido de encontrar a melhor solução, isto é, a solução ótima para um determinado critério de otimização. Devido à sua natureza, são métodos que conseguem encontrar as soluções ótimas, mas apenas para problemas de pequena dimensão (Madureira, 2003).

Os métodos exatos devem ser aplicados em problemas de pequena dimensão, pois para problemas de elevado grau de complexidade, a complexidade cresce exponencialmente com a dimensão do problema, pelo que estes algoritmos não conseguem encontrar um ótimo global num intervalo de tempo útil, requerendo também um grande esforço computacional. Assim, para problemas de pequena dimensão este tipo de métodos apresenta um bom desempenho. Os métodos exatos são particularmente importantes para a validação da eficácia obtida pelos procedimentos de aproximação (Pinedo, 2008).

Método *Branch and Bound*

Baseia-se numa enumeração implícita de todas as soluções do problema de otimização considerado. O espaço de pesquisa é explorado por construir dinamicamente uma árvore cujo nó, raiz, representa o problema que está a ser resolvido e todo o seu espaço de pesquisa associado e, os nós, folhas, são as possíveis soluções.

Este método assenta na divisão do espaço da solução (*branching*), cada divisão do espaço é distinta. Os limites (*bound*) da função objetivo são inferiores, se o objetivo for o de minimizar, ou superiores,

se o objetivo for maximizar. As ramificações são investigadas ao pormenor, com a finalidade de encontrar a solução ótima para o problema em questão, mas tendo sempre em conta os limites (inferiores ou superiores) estabelecidos e as ramificações que apresentarem soluções piores da encontrada são descartadas (Baker & Trietsch, 2009).

Utiliza uma árvore dinâmica para a representação do espaço de soluções de todas as sequências possíveis. Para iniciar uma pesquisa é aplicada uma partição e avaliação de uma sequência de nodos da árvore. Esta pesquisa inicia no nodo superior (raiz da árvore) e termina no último nível da árvore (folha) avaliado. Esta sequência de nodos resultante é considerada aceitável. Cada nodo de um nível p da árvore de procura representa uma sequência parcial de p operações (Agin, 1996).

O processo de *branch* (partição) consiste na escolha e divisão, partição, de um nodo ainda não selecionado no qual a procura irá avançar. O processo de *bound* (avaliação) elege a operação escolhida para continuar a procura. Em seguida, determina o limite inferior e o melhor limite superior alcançado, para o valor mínimo da função objetivo do nodo selecionado. Na maioria dos métodos *branch and bound*, o cálculo do limite superior inicial tem por base métodos heurísticos. Este cálculo é o primeiro passo a ser realizado antes do início do processo de procura. Se em algum nodo o limite inferior ultrapassar o valor do melhor limite superior, então não é necessário continuar essa procura nesse ramo, pois o melhor limite superior não vai melhorar mais. Neste caso, a procura recomeça no nodo anterior não visitado da árvore. O processo termina quando todos os nodos foram avaliados. O cálculo dos limites nos nodos é essencial para as técnicas *branch and bound*, pois evita a necessidade de se fazer cálculos para todo o espaço de soluções (Figueiredo, 2015). Na figura 1 encontra-se uma representação do método (El-Ghazali, 2009).

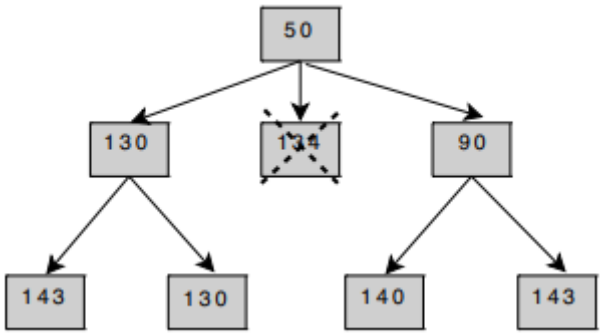


Figura 1 Representação do Método de *Branch and Bound* (El-Ghazali, 2009)

Na escolha do método *branch and bound* parte-se de uma solução, não inteira, dividindo o problema em ramificações. Cada ramificação representa uma solução parcial do problema original.

2.6.2. Métodos Aproximados

O recurso a métodos de aproximação é explicado pela complexidade dos problemas de escalonamento da produção e à escassez de tempo que se dispõe para a sua resolução. Em muitos casos não é possível determinar uma solução ótima. Estes métodos foram desenvolvidos para encontrar soluções para um problema, de uma forma simples e rápida, não garantido a melhor solução (Varela, 2007).

Segundo Morton & Pentico (1993), os métodos de aproximação tentam resolver cada instância de um problema de forma aproximada, não garantido a obtenção do ótimo global, mas permitem encontrar soluções satisfatórias que por norma são próximas da global. Contrariamente aos métodos exatos, esta classe de métodos pode ser aplicada a problemas de grande dimensão, uma vez que são rápidos e fáceis de implementar.

Heurísticas Construtivas

As heurísticas construtivas consistem na geração da solução utilizando regras ou procedimentos que determinam a ordem de processamento (sequência) das tarefas (French, 1982).

Estes métodos iniciam com uma solução vazia e expande iterativamente a solução atual até que uma solução completa seja construída (Cui et al., 2019). Isto é, são técnicas de otimização onde as soluções são construídas através de uma sequência de soluções de subproblemas. São denominadas técnicas míopes, pois na construção de uma solução não tomam em consideração o impacto das decisões em fases posteriores (Santos, 2015).

Como exemplos destas heurísticas temos as regras de prioridade (Pinedo, 2008), o algoritmo *Shifting Bottleneck* (Joseph Adams, Bals, & Zawack, 1988) e o algoritmo de *Giffler e Thompson* (Baker, 1974).

Pesquisa Local

A pesquisa local começa com uma solução inicial e em cada iteração a heurística substitui a corrente solução por um vizinho que melhora a função objetivo. Na figura 2 está representado o conceito de pesquisa local usando uma representação binária das soluções, com um operador de movimento inverso em que o objetivo é a maximização da função objetivo (El-Ghazali, 2009).

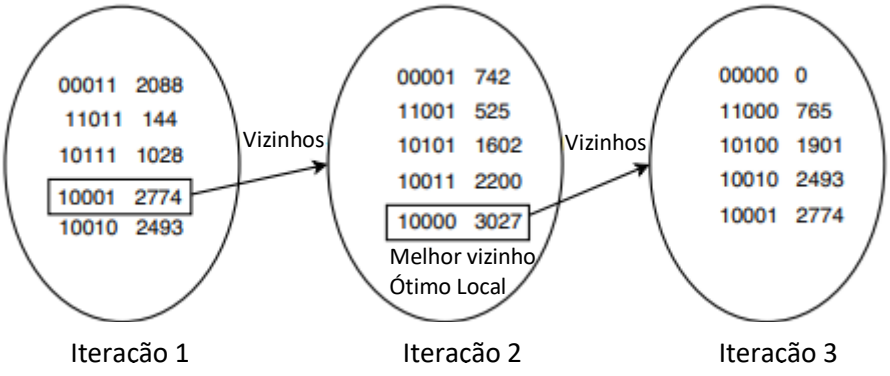


Figura 2 Esquema do conceito de Pesquisa Local (El-Ghazali, 2009)

A procura termina quando todos os vizinhos candidatos são piores que a solução corrente, isto significa que o ótimo local foi encontrado, e sendo que este método estuda a vizinhança local de uma dada solução completa.

O funcionamento básico de um algoritmo de pesquisa local consiste em partir de uma solução inicial, que pode ser gerada a partir de uma regra de prioridade. A seguir é gerada a vizinhança dessa solução, o objetivo do algoritmo é escolher a melhor solução dentro da vizinhança e fazer dela a solução principal. A forma mais simples de escolher uma nova solução consiste em selecionar uma que melhore o valor da função objetivo. O processo repete-se enquanto existir, dentro da vizinhança, uma solução que melhore a principal (Marques, 2015). Na figura 3 encontra-se uma representação do algoritmo (Madureira, 2009).

Algoritmo de Pesquisa Local
Início
$n \leftarrow 1$
Selecionar $x_1 \in F$
$x_1 \leftarrow solução\ inicial\ ( )$
$x' \leftarrow x_1$
$f' \leftarrow f(x_1)$
$N(x_1) \leftarrow gera\ vizinhança\ ( )$
$x_{n+1} \leftarrow solução\ melhor\ (N(x_1))$
Fazer
Se $f(x_{n+1}) \leq f(x')$ então
$x_n \leftarrow x_{n+1}$
$x' \leftarrow x_{n+1}$
$f' \leftarrow f(x_{n+1})$
Fim Se
$n \leftarrow n+1$
$N(x_n) \leftarrow gera\ vizinhança\ ( )$
$x_{n+1} \leftarrow solução\ melhor\ (N(x_n))$
Enquanto $f(x_{n+1}) \leq f(x_n)$
Fim

Figura 3 Representação do Algoritmo de Pesquisa Local (Madureira, 2009)

A escolha da solução inicial condiciona a qualidade do resultado final, e como tal para encontrar uma solução melhor através deste método deverá ser usado diferentes soluções iniciais.

As 3 estratégias mais comuns na seleção da vizinhança são:

- Melhor melhoria (*hill-climbing*): esta estratégia consiste em escolher o melhor vizinho, ou seja, o vizinho que mais melhora a função objetivo seja esta de minimização ou maximização. A vizinhança é avaliada de uma forma determinística. Consequentemente, a procura por novos vizinhos torna-se exaustiva, e todos os possíveis movimentos são experimentados. Este tipo de procura torna-se demorada para grandes conjuntos de vizinhos.

- Primeira melhoria: consiste em escolher o primeiro vizinho que tenha uma melhoria quando comparado com a corrente solução. Depois, o vizinho que melhora a solução é imediatamente selecionado para substituir a solução candidata. Esta estratégia envolve uma evolução parcial da vizinhança. Num movimento cíclico, a vizinhança é avaliada de maneira determinística, seguindo

uma determinada ordem de geração de vizinhos. Caso não seja encontrada nenhuma solução que permita melhora a função objetivo todos os vizinhos são considerados e avaliados.

-Seleção aleatória: esta estratégia é, como o nome indica, a seleção aleatória de um vizinho que melhore a solução atual (El-Ghazali, 2009).

2.7. Meta-Heurísticas

É cada vez mais frequente utilizar as meta-heurísticas para resolver os problemas mais complexos de otimização. A maioria das meta-heurísticas advêm do comportamento dos sistemas biológicos e/ou de sistemas físicos na natureza. A natureza evoluiu ao longo de milhões de anos e encontrou soluções para a maioria dos problemas que se surgiram. Podemos assim aprender do sucesso das soluções encontradas pela natureza para os problemas que defrontou e desenvolver heurísticas e meta-heurísticas baseadas nela (Yang, 2010a).

Os comportamentos das meta-heurísticas são avaliados pelos conceitos de intensidade e diversificação. O termo diversificação geralmente refere-se à exploração do espaço de procura, enquanto a intensidade é a exploração da experiência da pesquisa acumulada (pesquisa em regiões mais atrativas no espaço de soluções). Ou seja, diversificação significa a geração de diversas soluções como explorar o espaço de pesquisa numa escala global, enquanto intensidade é a focalização da procura numa região local (Blum & Roli, 2003).

O propósito das meta-heurísticas é produzir uma solução para um problema complexo num tempo admissível. A complexidade do problema de interesse torna possível a procura de cada solução possível ou combinação, o objetivo é encontrar uma solução plausível num espaço de tempo aceitável, mas não há garantia que a solução ótima seja encontrada. Portanto, o propósito é ter um algoritmo que permita atingir soluções com uma boa qualidade, ou seja, não há garantia que a solução ótima é atingida, mas as soluções encontradas são próximas desta (Yang, 2010b).

As meta-heurísticas podem ser baseadas em comportamentos populacionais ou de trajetória. Como exemplos de meta heurísticas baseadas em populações temos os algoritmos genéticos e o *Particle Swarm Optimization* (PSO). O primeiro é fundamentado pelo cruzamento de indivíduos da população de forma a encontrar indivíduos mais aptos e soluções com melhor desempenho. O PSO é baseado nos cardumes de peixes e nos bandos de pássaros (Sarangi, Samal, 2019).

O *Simulated Annealing* (SA), mais posteriormente abordado de forma pormenorizada, usa uma solução que se move através de espaço de soluções. Um movimento ou solução melhor é sempre aceite, enquanto que uma solução não tão boa pode ser aceite com uma certa probabilidade. Os movimentos traçam uma trajetória no espaço de soluções (Lopez & Roubellat, 2008).

As meta-heurísticas podem ser caracterizadas em 6 grupos (Gendreau & Potvin, 2010; Glover & Kochenberger, 2003; Osman, 2011):

- Algoritmos Evolucionários (AE): são procedimentos que tentam reproduzir a evolução das espécies. Especificamente estes algoritmos simulam os processos biológicos que permitem que as gerações consecutivas em uma população se adaptem ao eu ambiente. O processo de adaptação é aplicado principalmente pela herança genética de pais para filhos e pela sobrevivência do mais apto. Os algoritmos genéticos são os mais conhecidos deste tipo de método (Wu & Banzhaf, 1998)(Roshanzamir, Palhang, & Mirzaei, 2019).

- Baseados em Memória: a principal característica destas heurísticas é o uso da memória adaptativa e exploração responsiva. O papel da memória adaptativa é impedir que a pesquisa fique presa nas soluções ótimas locais e direcionar a pesquisa para processos de diversificação e intensidade mais efetivos. A heurística baseada em memória mais conhecida é a *Tabu Search* (Hedar, Mabrouk, & Fukushima, 2011).

- Pesquisa de Vizinhança: Parte de uma solução candidata e depois move-se para uma solução vizinha. No espaço de pesquisa são definidas uma relação e estrutura de vizinhança (Hansen & Mladenovic, 1999).

-*Swarm Intelligence*: são técnicas de inteligência artificial que estudam e simulam comportamentos coletivos e sistemas auto-organizados de exames de animais. Os mais estudados são o *Ant Colony Optimization* e o *Particle Swarm Optimization* (Engelbrecht, 2007).

-Baseado em Probabilidades: determinam se a solução atual é substituída por um novo ponto de avaliação com base numa probabilidade dependendo da diferença entre os seus valores da função. O *Simulated Annealing* é a mais conhecida (Hedar & Fukushima, 2002).

- Métodos Híbridos: Existem muitas possibilidades para compor as heurísticas híbridas. O uso de métodos de pesquisa local dentro de uma meta-heurística é a maneira mais eficaz de superar a morosidade. Além disso, a inicialização múltipla de um método de pesquisa local é outro esquema que compõe as meta-heurísticas híbridas. Como exemplo temos o *Memetic Algorithms* (Hedar & Fukushima, 2006).

O equilíbrio entre diversificação e intensidade permite identificar rapidamente regiões no espaço de pesquisa com soluções de alta qualidade e, por outro, não perder muito tempo nas regiões do espaço de pesquisa já explorado ou que não oferece alta qualidade de soluções. A estratégia de procura de cada meta heurística depende diretamente da filosofia intrínseca à mesma (Blum & Roli, 2003).

2.7.1. Simulated Annealing

Uma das meta-heurísticas mais antigas e mais populares é o *Simulated Annealing* (SA), foi introduzida por Kirkpatrick et al. (Kirkpatric, Gelatt Jr., & Vecci, 1983) como um método para resolver problemas combinatórios. Nos anos 80, o SA teve um grande impacto no campo das meta-heurísticas pela sua simplicidade e eficiência da solução para problemas de otimização combinatória. Só mais tarde, é que começou a ser usado em problemas de otimização contínua (Yang, 2010b).

Baseia-se nos princípios da mecânica estatística, por analogia com o processo de fundição (*annealing*) de um sólido, segundo os quais os processos de arrefecimento dos metais requerem o aquecimento a uma temperatura alta no qual o mesmo se funde e, em seguida, o arrefecimento lento para obter uma estrutura cristalina forte, até este se solidificar. A força da estrutura depende da taxa de arrefecimento do metal considerado (Marques, 2015).

Se a temperatura inicial não for suficientemente alta ou se for aplicado um arrefecimento rápido, são obtidas imperfeições (estados metaestáveis). Nesse caso, o sólido de arrefecimento não atingirá o equilíbrio térmico a cada temperatura. Cristais fortes são obtidos a partir de um

arrefecimento lento e cuidadoso. O algoritmo SA simula as mudanças de energia num sistema submetido a um processo de arrefecimento até convergir para um estado de equilíbrio.

A função objetivo do problema é análoga ao estado de energia do sistema. Uma solução do problema de otimização corresponde a um estado do sistema. As variáveis de decisão associadas a uma solução do problema são análogas às posições moleculares. O ótimo global corresponde ao estado fundamental do sistema. Encontrar um mínimo local implica que um estado metaestável foi atingido (El-Ghazali, 2009).

Desde o seu desenvolvimento que já foi aplicado em quase todas as áreas de otimização, contrariamente a métodos determinísticos que têm a desvantagens de ficarem presos a mínimos locais. A principal vantagem do SA é a sua capacidade de evitar ficar preso em ótimos locais, foi provado que irá convergir para o ótimo global se for usado aleatoriedade combinado com um arrefecimento muito lento (Fleischer & Jacobson, 1999).

O SA é um algoritmo estocástico que permite, em algumas condições, a degradação de uma solução. O objetivo é escapar dos ótimos locais e, assim, atrasar a convergência. É uma meta-heurística sem memória no sentido que o algoritmo não usa nenhuma informação obtida durante a pesquisa, parte de uma solução inicial aleatória e gera em cada iteração uma nova solução através de um mecanismo de vizinhaça. Os movimentos que melhorem a função objetivo são sempre aceites, caso contrário, o vizinho é selecionado com uma determinada probabilidade que depende da temperatura atual e da diferença entre a função objetivo da corrente solução e a solução gerada pela vizinhança. À medida que o algoritmo avança, a probabilidade de novas soluções serem aceites diminui.

Usa um parâmetro de controlo, a temperatura, que ajuda a determinar a probabilidade de aceitar soluções que não melhoram a função objetivo. Uma vez atingido o estado de equilíbrio, a temperatura diminui gradualmente. Desta forma, no início muitos movimentos são permitidos, mas com o avançar do algoritmo apenas os que permitem melhorias na função objetivo é que são escolhidos (Low, 2005).

Na figura 4 encontra-se uma representação do funcionamento do *Simulated Annealing.* Quanto mais alta a temperatura, maior probabilidade de aceitar soluções piores. A uma determinada temperatura, quanto menor o aumento da função objetiva menor é a probabilidade de aceitar as soluções piores. É importante relembrar que soluções que apresentem uma melhor função objetiva são sempre aceites (El-Ghazali, 2009).

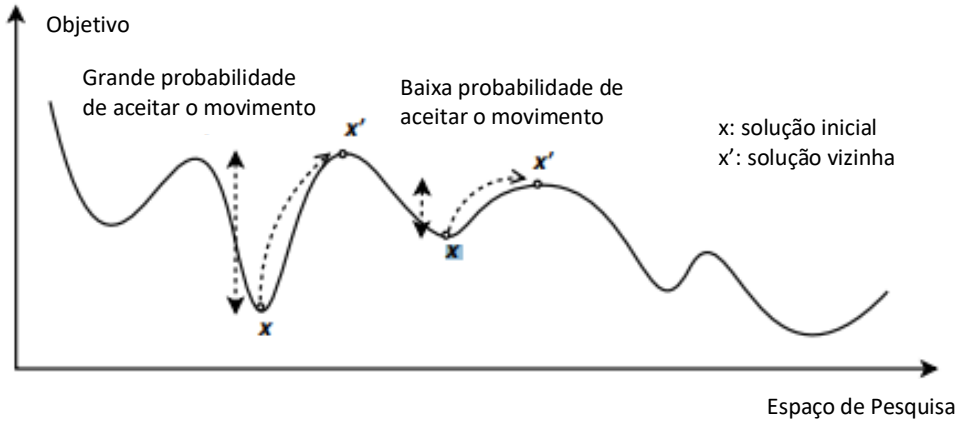


Figura 4 Representação do funcionamento do Simulated Annealing (El-Ghazali, 2009)

O objetivo deste método é encontrar a melhor solução de entre um conjunto de soluções possíveis, uma das vantagens é permitir encontrar soluções próximas do ótimo global sem ser necessário um esforço computacional elevado. É importante referir que neste tipo de métodos não é possível saber se a solução encontrada é um ótimo global ou um ótimo local (Varela, 2007).

Na figura 5 apresenta-se o esquema representativo do algoritmo do SA.

Algoritmo *Simulated Annealing*
Entrada: Escalonamento
$s = s_0$; /*Geração da solução inicial */
$T = T_{max}$; /*Temperatura Inicial */
Repetir:
Repetir: /*Até uma dada temperatura */
Gerar um vizinho aleatório x'
$\Delta E = f(x') - f(x)$
Se $\Delta E \leq 0$ então $s = s'$ /* Aceita a solução do vizinho */
Senão: Aceita s' com uma dada probabilidade
Até: o número de iterações (L)
$T = g(T)$; /* Atualização da Temperatura */
Até: O critério de paragem ser satisfeito
Output: Melhor Solução Encontrada

Figura 5 Algoritmo do Simulated Annealing (El-Ghazali, 2009)

Parametrização do Simulated Annealing

O *Simulated Annealing* inicia com uma solução (X), uma temperatura inicial (T), um número de iterações (L) e uma constante α ou β entre zero e um, que representam um de dois esquemas de arrefecimento. A temperatura (T) controla a possibilidade de aceitação de uma solução deteriorada e um número de iteração (L) decide o número de repetições até que a solução atinja um estado estável sob a temperatura. Para uma temperatura alta existe a flexibilidade e, maior probabilidade de avançar/encontrar uma solução pior, mas conforme se vai decrescendo a temperatura existe menos essa flexibilidade e probabilidade (Kim, Kim, Jang, & Chen, 2002).

Uma das particularidades do SA é, como referido anteriormente, que uma nova solução, não precisa de ter um melhor desempenho para ser aceite, ou seja, para cada nova solução é calculada uma probabilidade dada pela **Erro! A origem da referência não foi encontrada.**0, onde x_1 representa a nova solução e x a melhor solução encontrada até ao momento, que posteriormente é comparada com uma solução aleatória para ser aceite ou não.

$$P = e^{\left(-\frac{f(x_1)-f(x)}{T}\right)} \quad 10$$

O esquema de arrefecimento define a temperatura do algoritmo em cada iteração, pelo que tem um papel importante no seu sucesso, pois tem de existir um compromisso entre a qualidade da solução e a velocidade do cronograma de arrefecimento. Se a temperatura diminuir lentamente

são obtidas soluções melhores, mas com um tempo de computação mais significativo. A temperatura (T) pode ser atualizada da seguinte forma (El-Ghazali, 2009):
-Linear: na programação linear trivial, a temperatura T é atualizada pela seguinte formula:

$$T_i = T - \beta \quad (11)$$

Onde T_i representa a temperatura em cada iteração i.
Onde β é um valor constante entre]0,1[.
-Geométrico: na programação geométrica, a temperatura é atualizada por:

$$T = \alpha \times T \quad (12)$$

O fator α está entre]0,1[. Por norma é o esquema de arrefecimento que apresenta mais robustez e é referido que o valor deve estar entre 0,50 e 0,99 (El-Ghazali, 2009)(Marques, 2015).
-Logarítmico: a fórmula usada para este esquema de arrefecimento é a seguinte:

$$T_i = \frac{T_0}{\log(i)} \quad (13)$$

Este esquema é muito lento para ser aplicado na prática, mas tem a propriedade de convergir para um ótimo global.
Por último temos o número de iterações (L). Este parâmetro, normalmente, é definido em função da dimensão do problema. Ou seja, num problema de sequenciamento corresponde ao número de tarefas consideradas(Santos, 2015)(M.-W. Park & Kim, 1998).

3. DESCRIÇÃO DO PROBLEMA E IMPLEMENTAÇÃO COMPUTACIONAL

Esta secção tem como objetivo apresentar o problema inerente ao objetivo que se pretende alcançar, assim como a apresentação do método de resolução capaz de automatizar e otimizar a programação da produção.

Tendo em conta os objetivos traçados para este capítulo, ele encontra-se dividido em três subcapítulos nomeadamente a descrição do problema, que como o nome indica é a explicação e fundamentação do problema de escalonamento da produção. A implementação computacional que diz respeito a ferramenta de apoio de decisão, que permite apresentar um método de resolução do problema encontrada através da meta-heurística selecionada. Por fim, a parametrização da meta-heurística escolhida, particularmente o *Simulated Annealing*, sendo definido os intervalos dos parâmetros da mesma e as escolhas efetuados para a realização dos testes que serão abordados no capítulo seguinte (Capítulo 4).

3.1. Descrição do Problema

A abordagem que se segue centra-se na pormenorização do problema, assim como todas as implicações que este poderá ter numa empresa. As empresas estão sujeitas às várias flutuações do mercado, por isso vivenciam novos problemas, os quais têm de ser identificados e resolvidos para que a empresa tenha uma melhor eficácia e, com isso, menor despesa e maior lucro.

Os mercados são cada vez mais voláteis, isto é, a rapidez com que a procura aumenta e diminui é cada vez mais frequente. Sendo assim as organizações têm de estar permanentemente a tomar decisões tanto a nível produtivo como ao nível dos recursos humanos. A fase de planeamento da produção que afeta diretamente todos os recursos da empresa, desde os recursos humanos até aos recursos máquina, tem uma extrema importância pois define a eficiência dos seus recursos. Comprometendo a sua eficiência afeta diretamente a sua competitividade, traduzindo-se numa perda de vendas e quota de mercado.

É exigido as empresas cada vez mais uma maior variedade de artigos de produção, que leva ao aumento exponencial da complexidade da gestão da produção. A eficiência dos seus recursos é um fator chave de competitividade de uma organização, o que leva com que a gestão da produção seja cada vez mais importante. Assim, o que produzir, quando e em que quantidades diz respeito à programação da produção, tornando esta atividade uma das mais importantes para o seu sucesso.

Programação da produção corresponde a afetação de operações, equipamentos, materiais e mão de obra à capacidade disponível, assim como definir o lançamento e sequência de operações de forma a garantir os prazos de entrega (Cruz, 2018). Visa a programação da execução de um conjunto de trabalhos num conjunto de recursos, máquinas, durante um determinado intervalo de tempo (Varela, 2007). Através do escalonamento existe manobra para a competitividade onde o propósito é a satisfação de objetivos definidos ou preferências de escalonamento individuais.

Para além de toda a volatilidade dos mercados existe também todos os acontecimentos do dia a dia, avaria de máquinas, falta de recursos humanos, dependência dos fornecedores, entre outros.

Qualquer que seja o setor, a organização depende sempre dos seus fornecedores, mesmo quando o fornecedor faz parte do mesmo grupo que a empresa. Depende das entregas atempadas e sempre que ocorre um atraso na entrega condicionada toda a continuidade dos trabalhos. Se as empresas dependerem de terceiros fora da sua área de fabricação, fora do país da localização fabril tem tendência a agravar mais.

Por outro lado, as avarias por vezes são bastante complicadas visto que se tem de conseguir entender a causa de um determinado efeito visível. Normalmente têm-se problemas em encontrar a causa de um problema. Todos estes motivos levam a uma necessidade obrigatória, em que tudo tem de ser realizado no menor tempo possível, de forma a garantir uma margem de erro, para as possíveis falhas, evitando, minimizando o impacto dos atrasos.

Tendo em conta todas estas adversidades que podem ocorrer e o cumprimento dos prazos apertados exigidos pelos clientes, a gestão da produção tem de responder rapidamente a todas as alterações iminentes dai ser extremamente importante automatizar a função.

O contexto do problema que se pretende minimizar é caracterizado por um ambiente onde existe uma rede de produção e a distribuição de diferentes trabalhos, que podem ser considerados produtos finais. Cada produto/trabalho tenho uma sequência de máquinas especifica pelo qual têm de ser processado. Os trabalhos são ligados pelas relações de precedência temporal.

Neste trabalho de investigação o escalonamento da produção envolve a afetação, decisão e calendarização e sequenciação dos trabalhos. A afetação é quando a mesma tarefa pode ser executada em vários postos de trabalho. Geralmente a calendarização compreende decisões que sucedem a afetação, sendo que há a decisão de distribuir as operações no tempo, de forma a otimizar uma medida de desempenho, determinando quando as tarefas vão ser executadas. No mesmo sentido, a sequenciação determina a ordem de execução dos trabalhos, independentemente de os tempos de execução serem difusos.

Devem ser descritas várias experiências de resolução, de forma a perceber, com aplicação de cada uma, qual a que alcançará um resultado mais otimizado consoante a medida de desempenho. A aplicação de um só método poderá levar a uma solução que torne o processo de fabrico mais rápido do que o atual, dado não estar estabelecido qualquer tipo de escalonamento, mas mesmo assim corremos o risco de não otimizar ao máximo todo o processo. Assim através da comparação das várias soluções poder-se-á chegar a um resultado, que deverá ser considerado o de melhor aplicação.

O problema considerado para o desenvolvimento desta ferramenta de apoio à tomada de decisão é o problema em J*ob-Shop*, sendo o mais comum nas empresas atualmente. Os problemas em J*ob--Shop* são descritos por um conjunto de trabalhos e um conjunto de máquinas, em que cada trabalho consiste em uma cadeia de operações, cada uma das quais precisa de ser processada durante um período e, esse tempo não pode ser interrompido. Cada máquina pode processar no máximo uma operação de cada vez.

A escolha de problemas em J*ob-Shop* prendeu-se, também, por ser um dos problemas que permite a modelação de qualquer outro problema de escalonamento e, pela sua complexidade de resolução. Sendo que, os problemas em J*ob-Shop* se concentram na determinação da permutação dos trabalhos em cada máquina sujeitas às restrições de processamento, com a finalidade de atingir

o objetivo traçado são considerados problemas NP-hard, ou seja, problemas difíceis de obter o ideal global por algoritmos exatos, mesmo quando a sua gama é pequena (Jiang & Zhang, 2018).

Surgem as meta-heurísticas como uns dos principais métodos de otimização dos problemas de programação da produção. Pesquisam a totalidade do espaço de soluções do problema, isto é, são dotadas de mecanismos estocásticos que permitem obter maior diversidade na pesquisa do espaço de soluções e ultrapassar os ótimos locais.

Considerando as técnicas abordadas no capítulo 2, onde estão presentes os fundamentos teóricos, de forma a enquadrar o problema e definido o melhor método para a resolução do problema a aplicação da meta-heurística *Simulated Annealing* por apresentar a vantagem de conseguir lidar com modelos não lineares, dados desordenados e com restrições, para além se ser bastante robusto permite convergir para a solução ótima (Loureiro, 2014).

3.2. Implementação Computacional

O objetivo deste projeto é desenvolver um método de resolução capaz de automatizar e otimizar o escalonamento da produção para combater os problemas enunciados anteriormente. Tendo em conta os trabalhos que se pretendem realizar e as sequências destes nas máquinas, é utilizada uma meta-heurística que satisfaz da melhor maneira a minimização do *makespan*, isto é, a minimização do somatório dos tempos de processamento.

A proposta de trabalho envolve o desenvolvimento de uma interface para auxiliar a tomada de decisão por parte dos gestores de produção. O protótipo aplica a meta-heurística *Simulated Annelaing,* anteriormente descrito no capítulo *2.7.1*, em qualquer tipo de problema de escalonamento que o utilizador pretenda resolver.

Sendo que o programa é capaz de resolver um problema de Job-Shop, isto significa que ele é capaz de resolver todos os tipos de problemas de escalonamento, nomeadamente, os problemas em máquina única, em máquinas paralelas, Flow-Shop e Open-Shop.

A escolha da meta heurística SA utilizada na implementação assenta no facto de esta meta-heurística convergir sempre para uma solução ótima, aquando de uma minuciosa escolha dos seus parâmetros combinado um arrefecimento muito lento (Loureiro, 2014).

Para os problemas para o qual se pretende uma otimização assume-se:

- todos os trabalhos (j) estão disponíveis no momento zero;
- cada uma das máquinas só pode processar um trabalho de cada vez;
- não são permitidas interrupções dos trabalhos;
- a cada trabalho está associado um tempo de processamento (p);

O objetivo passa por determinar a sequência de operações nas máquinas de forma a respeitar todas as restrições impostas e otimizar a instância testada no algoritmo que é a soma dos tempos de processamento. A ferramenta foi desenvolvida pela linguagem de programação C#, recorrendo ao *Visual Studio*.

O layout da interface inicial é representado na figura 6.

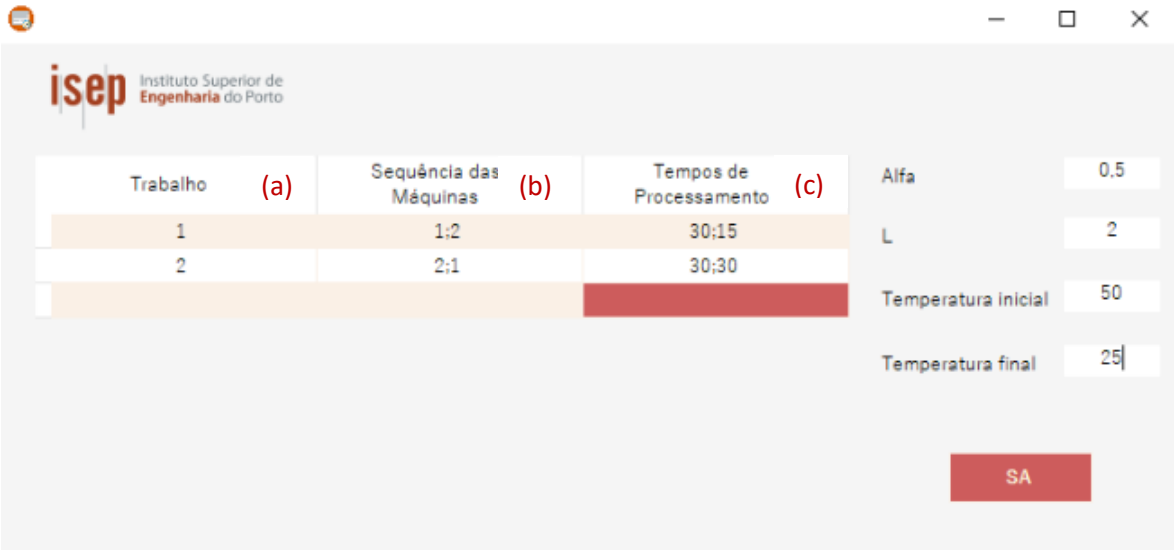


Figura 6 Layout da interface inicial

A introdução dos dados é feita pelo esquema apresentado na tabela 1. Em (a) o utilizador coloca os trabalhos que pretende que sejam escalonados, enquanto que em (b) introduz a sequência das máquinas pelo qual o trabalho respetivo tem de passar para ser concretizado, separados por ponto e vírgula (;) e por último em (c) insere o tempo de processamento correspondente ao trabalho a ser executado na respetiva máquina, novamente separados por ponto e vírgula (;).

Tabela 1 - Exemplo da introdução dos dados

Trabalho	Sequência das Máquina	Tempos de Processamento
1	1;2	30;15
2	2,1	30;30

O botão diz respeito ao SA. Para poder correr a meta-heurística o utilizador terá de preencher, para além da tabela de introdução dos dados, as caixas de texto correspondentes aos parâmetros do algoritmo. Os parâmetros são os característicos: L (números de iterações), temperatura inicial, alfa, que vai permitir o decréscimo da temperatura até atingir uma temperatura final, também inserida pelo utilizador. Se faltar algum destes parâmetros, o SA não será corrido dando uma mensagem de erro como o exemplo da figura 7.

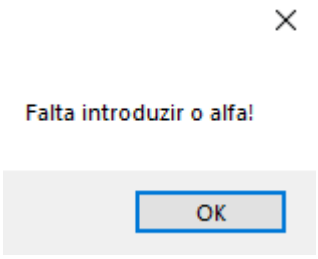


Figura 7 - Mensagem de erro pela falta de introdução do alfa

A solução é obtida como já referido pela minimização da medida de desempenho *makespan.*

Na figura 8, está representado a segunda interface que corresponde a tabela da solução ótima e o *makespan* obtido para essa mesma solução.

Figura 8- Interface da solução ótima

Nessa mesma interface há um botão denominado de “Gantt” que permite a visualização da solução ótima apresentada na tabela, mas agora num gráfico de Gantt.

Por fim, na figura 9, é representado o diagrama de *Gantt* da solução calculada. O gráfico de *Gantt* permite a fácil interpretação da solução, sendo que o eixo das ordenadas representa as máquinas e o eixo das abcissas representa o tempo, por fim os trabalhos são representados pelos retângulos em que cada cor tem a correspondência identificada na tabela apresentada abaixo.

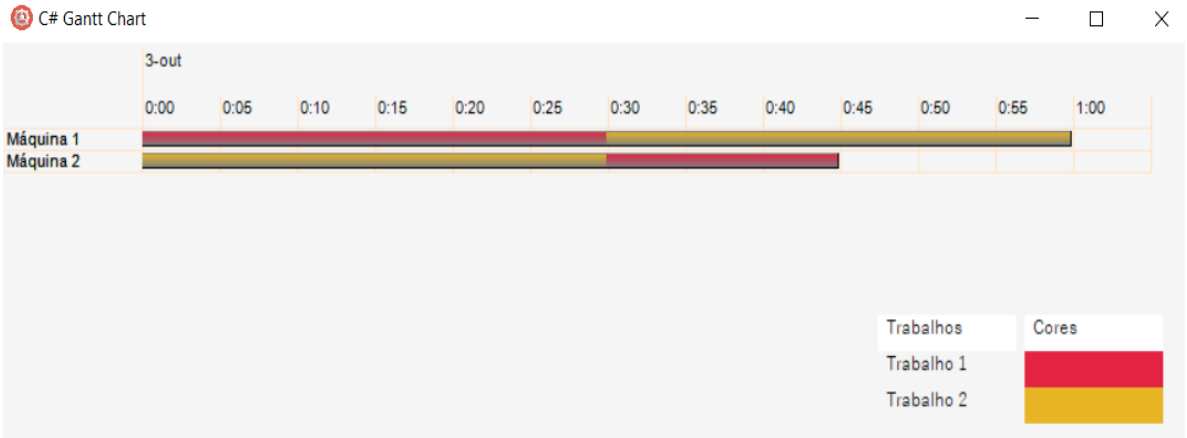


Figura 9 Interface do gráfico de Gantt

É importante referir que os dados introduzidos na ferramenta e demonstrados na tabela 1 são apenas um exemplo e em nenhuma parte corresponde a uma realidade

O próximo passo para a terminação deste projeto será a validação da ferramenta através de problemas adequados e a apresentação dos resultados com fundamentos estatísticos.

A sequenciação dos passos a seguir na utilização da ferramenta encontram-se descritos na tabela 2:

Tabela 2 Sequenciação dos passos de utilização da ferramenta de apoio á decisão

Passo	Descrição
1	Introdução para cada trabalho a sequência das máquinas e os respetivos tempos de processamento
2	Introdução dos Parâmetros do *Simulated Annealing*
3	Quando a solução estiver calculada irá surgir uma nova interface que apresenta uma tabela com a descrição dos trabalhos em cada máquina
4	Permir o botão "*Gantt*"
5	Nova interface com o gráfico *Gantt* da solução encontrada

Para o funcionamento do protótipo considerou-se uma implementação modular, por permite a introdução de novos módulos quando for necessário, isto é, se for necessário acrescentar uma nova medida de desempenho ou uma nova meta heurística, será mais fácil a sua implementação através deste método.

Na figura 10 está representado o esquema modular que constitui o programa. Através da análise do esquema é possível afirmar que após a introdução dos dados pelo utilizador o programa segue o esquema pormenorizado na figura 10. Primeiramente corre o módulo Recolha de Dados ( ), que como o nome indica consiste na impressão dos dados do problema num *array*. Seguidamente no módulo SA ( ) determina-se se o problema a correr é um problema *Job-Shop* ( ), *Flow-Shop* ( ), Máquinas Paralelas ( ) ou ainda Máquina Única ( ). Se o problema for caracterizado pelo tipo *Job--Shop* será calculado uma solução inicial no módulo Movimento1 ( ), a avaliação desta solução, e de todas as outras que serão geradas através do Módulo Estrutura de Vizinhança ( ), é feita no módulo Avaliar1 ( ). A mesma lógica se mantém para os problemas que não são *Job-Shop* apenas com a diferença que todas as máquinas terão a mesma sequência dos trabalhos.

Quando o algoritmo SA ( ) terminar irá transmitir a solução ótima encontrada de novo para o módulo Recolha de Dados ( ) e este permitirá o aparecimento gráfico da solução ótima no módulo Apresentação da Solução ( ). Por último, temos o módulo *Gantt* que permite a impressão do gráfico da *Gantt* da solução ótima.

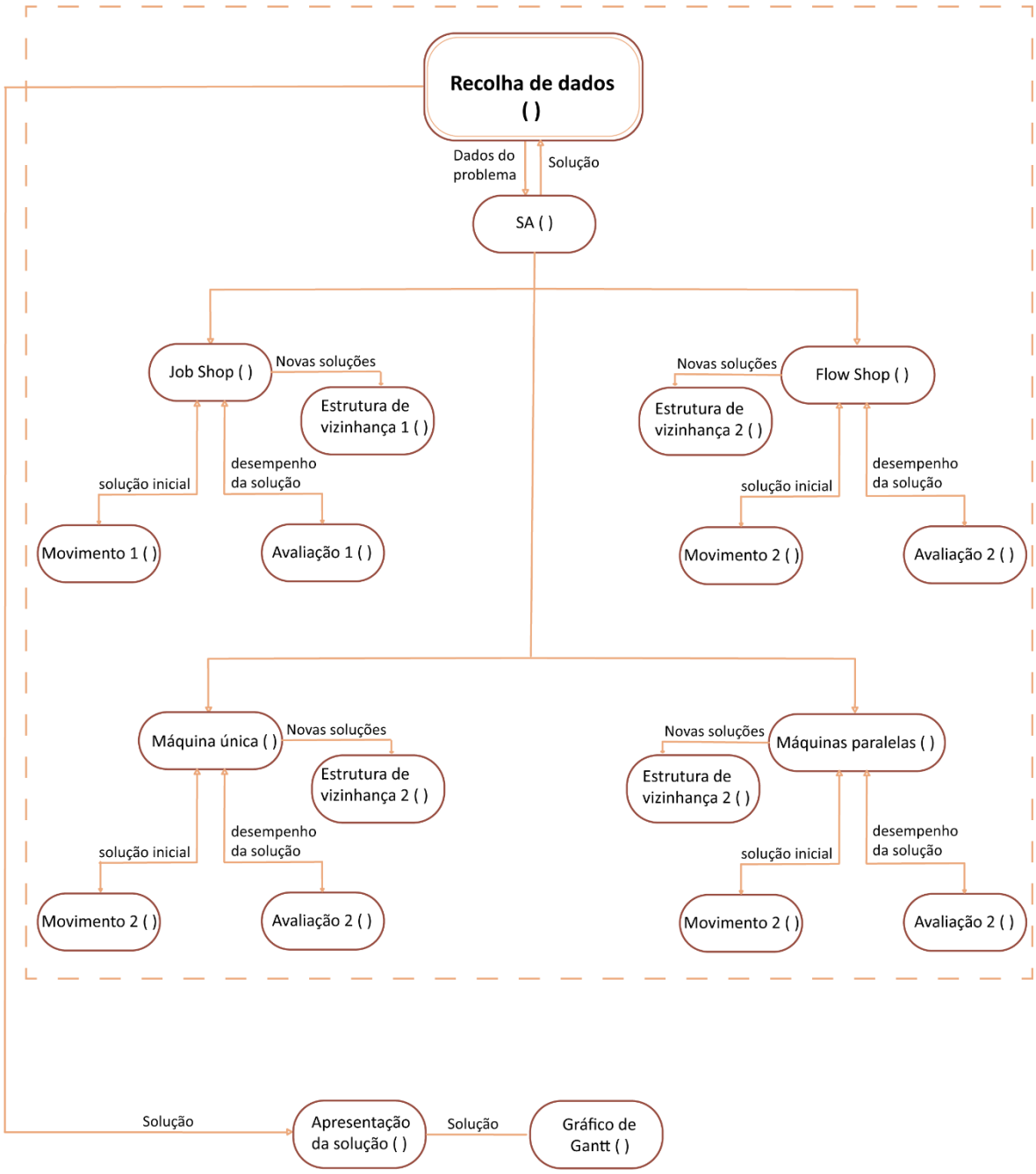


Figura 10 Esquema Modular do Protótipo

3.3. Parametrização do *Simulated Annealing*

Uma das vantagens do *Simulated Annealing* é o facto de não necessitar de um processo de parametrização detalhado, isto é, se a temperatura diminuir lentamente a meta-heurística vai encontrar um ótimo global num número infinito de iterações (Fleischer & Jacobson, 1999) ou se for inicializado com uma temperatura suficientemente alta e for diminuindo muito lentamente, o SA vai ter também um bom desempenho (Anily & Federgruen, 1987) (M.-W. Park & Kim, 1998).

Geralmente há uma limitação em termos de tempo para encontrar uma resolução para o problema de escalonamento apresentado e nesses casos é necessário determinar com precisão os parâmetros do SA de maneira a maximizar o desempenho e a eficiência do algoritmo (Santos, 2015).

Com já foi referido os parâmetros do SA são: a temperatura inicial, o número de iterações, o esquema de arrefecimento e o critério de interrupção.

Segundo 104, a Temperatura Inicial deverá ser um valor que permita uma razão de aceitação inicial entre [0.7,0.8] e de acordo com esta linha de pensamento Park e Kim (1998) recomenda uma probabilidade de aceitação de piores soluções próxima de 1. Eglese (1990) e Rose (1990) propôs a seguinte equação:

$$T_i = \frac{\Delta f(x)}{\ln(P_i)} \qquad 14$$

P_i corresponde a probabilidade de aceitação e Δf(x) corresponde á dispersão da amostra.

Tendo em conta que o esquema de arrefecimento utilizado nesta implementação computacional é o geométrico são sugeridos valores de α entre [0.80,0.99] em (Jonathan Rose, Klebsch, & Wolf, 1990). Mas segundo Cho (2005) são indicados valores entre [0.50,0.99]. É apresentado também uma forma de calcular α, em T_i representa a temperatura inicial, T_f a temperatura final e M o número de patamares de temperatura (M. W. Park & Kim, 1998).

$$\propto = \left(\frac{T_f}{T_i}\right)^{\frac{1}{(M-1)}} \qquad 15$$

Relativamente ao número de iterações, L, que define o número de soluções que são aceites até a temperatura diminuir. Segundo Laarhoven e Arts (1987) o número de iterações deverá ser do tamanho da instância do problema, isto é, num problema de sequenciamento corresponde ao número de tarefas que estão a ser consideradas no mesmo (André Santos, 2020).

3.4. Codificação

Para que seja possível o SA calcular soluções é necessário codificar ou representar o problema. A codificação tem de ser completa, ou seja, permite representar qualquer solução do problema, contactável, isto é, a sua manipulação (pela estrutura de vizinhança mais a frente apresentada) deve permitir encontrar qualquer solução do problema e ser eficiente, por outras palavras, deslocar-se rapidamente para as regiões mais promissoras do espaço de soluções (El-Ghazali, 2009).

A codificação selecionada para a implementação deste protótipo consiste numa codificação de permutação, o que indica a sequência de processamento. Ou seja, uma solução {4,2,1,3}, sequência

de execução na máquina correspondente será primeiro a tarefa 4, seguida da tarefa 2, posteriormente a tarefa 1 e por último a tarefa 3. Assim, esta codificação respeita os procedimentos anteriormente identificados e permite a aplicação da estrutura de vizinhança identificada no próximo capítulo.

3.5. Estrutura de Vizinhança

A estrutura de vizinhança é um ponto fundamental na implementação da Meta-Heurística, se a estrutura não for adequada promete a resolução do problema, pois a pesquisa pode não ser suficiente para percorrer o espaço de soluções. Pode ser definida como o conjunto de movimentos a partir do qual cada uma das soluções do problema é gerada (El-Ghazali, 2009).

O mecanismo de vizinhança pode ser visualizado na figura 11, onde se depreende que, após a geração aleatória da solução inicial, o conjunto de soluções posteriormente geradas advêm da troca aleatória de duas posições. É possível analisar na figura 12 um exemplo da estrutura de vizinhança considerada, onde foi gerado dois números aleatórios (4 e 2) que correspondem as posições que são trocadas entre si e um outro número aleatório que diz respeito á máquina que vai sofrer as alterações.

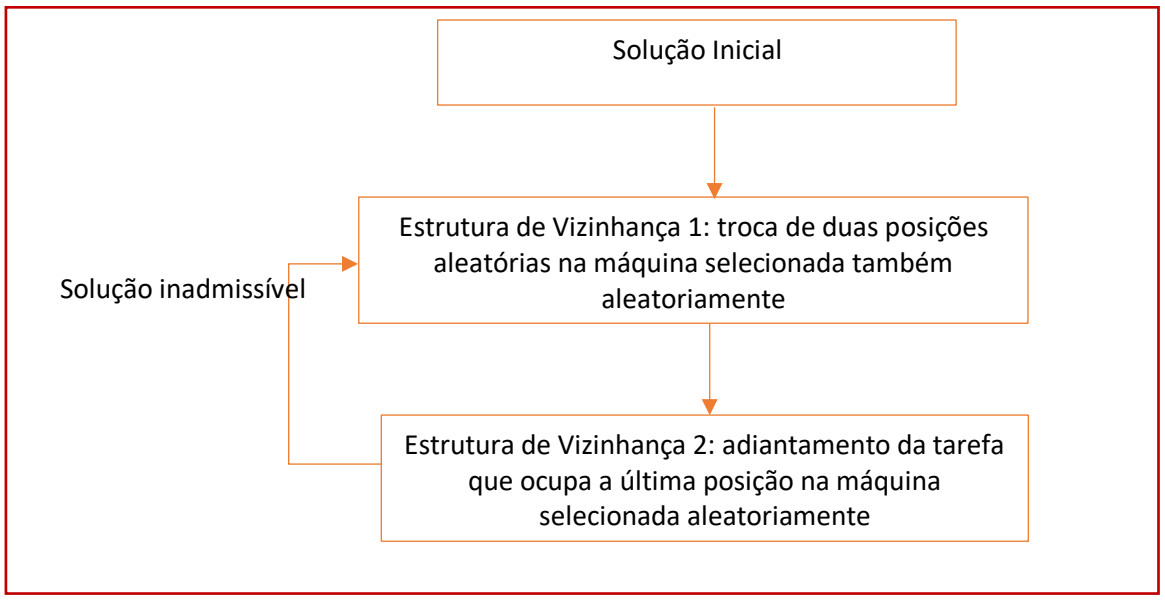


Figura 11 Representação das Estruturas de Vizinhança

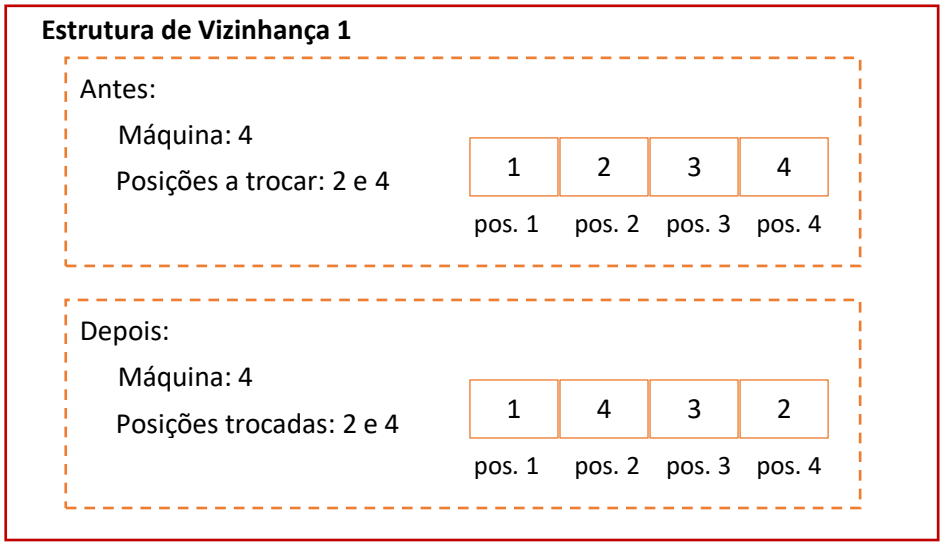


Figura 12 Representação da Estrutura de Vizinhança 1

Ainda relativamente á análise da figura 11, pode-se afirmar que quando a solução gerada pelo meio da estrutura de vizinhança 1 é inadmissível, gera-se uma nova solução pela escolha aleatória de uma máquina no qual se experimenta adiantar a última tarefa até a solução ser admissível (Figura 13). Se após todas as trocas possíveis pela estrutura de vizinhança 2 a solução continuar inadmissível é novamente gerada uma nova solução pela estrutura de vizinhança 1.

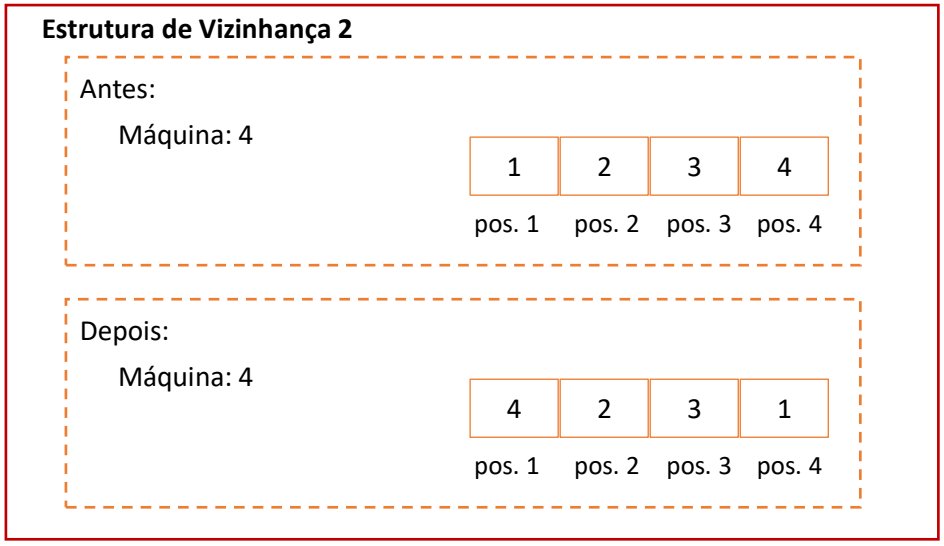


Figura 13 Representação da Estrutura de Vizinhança 2

3.6. Instâncias de Teste

O estudo computacional irá ser realizado divido em dois tipos de problemas de implementação Industrial. Primeiramente para implementações industriais do tipo F*low Shop*. Podem ser encontradas na *ORLibrary* em: (Taillard, 1993).

O recurso a problemas académicos permite conhecer a posterior as soluções ótimas, podendo ser usadas para análise de resultados apresentada no capítulo seguinte. Na tabela 3 é possível visualizar as instâncias de flow shop desenvolvidas por Taillard e a caracterização das mesmas com a sua respetiva solução ótima (Taillard, 1989).

Tabela 3 Instância de Teste *Flow Shop*

Instância	Nr. de Tarefas	Nr. de Máquinas	Solução Ótima
1	20	5	1232
2	20	5	1290
3	20	5	1073
4	20	5	1268
5	20	10	1448
6	20	10	1479
7	20	10	1407
8	20	10	1308
9	20	20	1911
10	20	20	1711

Para os problemas de Job Shop foram também usadas instâncias recorrendo novamente à libraria *ORLibrary* mas desta vez uma libraria apropriada para este tipo de problemas constituída por diversos problemas com tamanhos diferenciados de aplicação (Mattfeld & Vaessens, 1992). Na tabela 4 apresenta-se a dimensão dos problemas testados e as suas soluções ótimas (J. Adams, Balas, & Zawack, 1988)(Fisher & Thompson, 1963)(Lawrence, 1984)(D. Applegate, 1991)(Storer, Wu, & Vaccari, 1992)(Yamada & Nakano, 1992).

Tabela 4 Instâncias de Teste *Job Shop*

Instância	Nr. de Tarefas	Nr. de Máquinas	Solução Ótima
FT06	6	6	55
ORB01	10	10	1059
ABZ05	10	10	1234
LA19	10	10	847
SWV01	20	10	1518
SWV04	20	10	1642
ABZ07	20	15	656
SWV06	20	15	1913
SWV09	20	5	1918
YN01	20	20	967

Tendo em conta o capítulo 3.3, onde se encontra definido a parametrização do SA os parâmetros definidos para correr as instâncias encontram-se representados na tabela 5:

Tabela 5 Parâmetros definidos paras as Instâncias

	Alfa	L	Temperatura Inicial	Temperatura Final
1	0,99	2000	2000	1
2	0,995	6000	6000	1
3	0,99	7000	6500	1

4. RESULTADOS E DISCUSSÃO

Esta secção divide-se em 3 partes. A primeira é caracterizada pela apresentação dos resultados para cada instância com os respetivos parâmetros anteriormente definidos.

O seguinte subcapítulo diz respeito a análise dos resultados através da análise estatística, onde são apresentados os testes de hipóteses.

Por último, temos o subcapítulo da discussão dos resultados onde está exposto a relevância dos resultados para a questão de investigação.

4.1. Apresentação de resultados

As experiências computacionais foram realizadas num OMEN by HP, com um processador Intel Core i5 de 2,40GHz, com 8,00 GB de RAM e um sistema operativo Windows 10.

Os resultados das instâncias de *Flow-Shop* podem ser observados na tabela 6, onde é possível constatar a dimensão de instância, a melhor solução ótima encontrada para cada repetição com os diferentes parâmetros anteriormente definidos e a solução ótima da literatura, também anteriormente definida. Na tabela N representa o número de tarefas e M o número de máquinas.

Tabela 6 Resultados Obtidos nas Instâncias de *Flow-Shop*

Instância	N x M	SA	Solução Ótima
1	20 x 5	1247	1232
2	20 x 5	1292	1290
3	20 x 5	1097	1073
4	20 x 5	1304	1268
5	20 x 10	1493	1448
6	20 x 10	1498	1479
7	20 x 10	1431	1407
8	20 x 10	1366	1308
9	20 x 20	2020	1911
10	20 x 20	1813	1711

Como é possível observar, as soluções obtidas estão muito próximas da solução ótima para todas as instâncias através do SA.

Na tabela 7 apresenta-se agora os resultados obtidos para as instâncias de *Job-Shop*, nomeadamente os problemas FT06, ORB01, ABZ05, LA19, SWV01, SWV04, ABZ07, SWV06, SWV09, YN01. Nesta tabela encontra-se a melhor solução para cada um dos parâmetros testados para o *Simulated Annealing*

Tabela 7 Resultados Obtidos para as Instância *Job-Shop*

		Parâmetros			
Instância	N x M	1	2	3	Solução Ótima
FT06	6 x 6	60	57	55	55
ORB01	10 x 10	1176	1125	1092	1059
ABZ05	10 x 10	1270	1257	1250	1234
LA19	10 x 10	863	863	854	847
SWV01	20 x 10	1641	1641	1575	1518
SWV04	20 x 10	1801	1783	1748	1642
ABZ07	20 x 15	670	668	668	656
SWV06	20 x 15	2251	2104	1971	1913
SWV09	20 x 5	2048	2048	2024	1918
YN01	20 x 20	1357	1127	1126	967

Para os problemas do tipo *Job-Shop* as soluções encontradas pioraram relativamente ao ótimo.

É possível verificar na figura 14 o desempenho do SA relativamente á solução ótima para o problema de *Flow-Shop*.

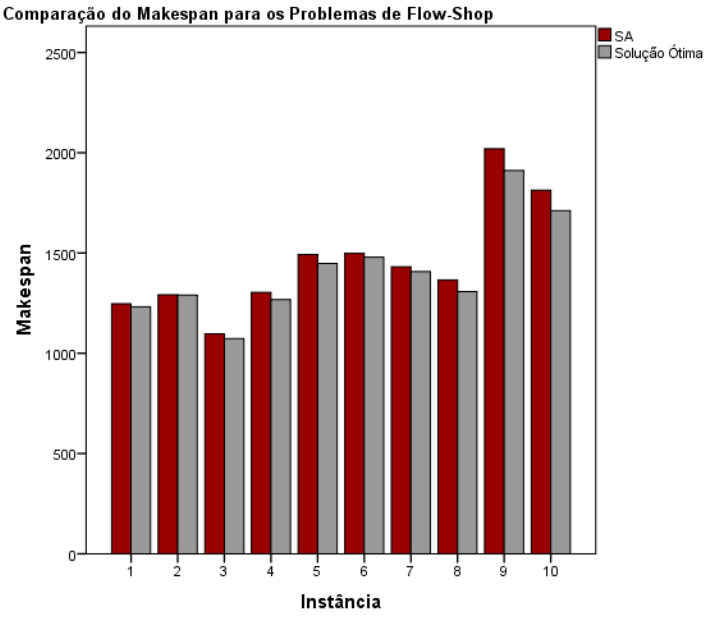


Figura 14 Makespan do SA comparativamente á Solução Ótima para o problema de *Flow-Shop*

Para o problema de *Job-Shop* também se desenvolveu o mesmo gráfico de comparação do desempenho do SA relativamente á solução ótima, sendo que foi considera a melhor solução outra de todos os parâmetros desenvolvidos.

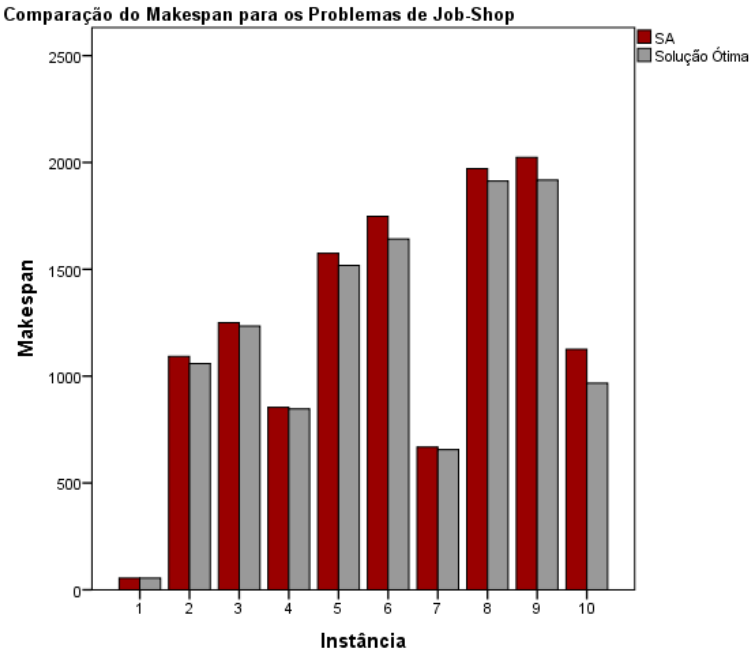


Figura 15 Desempenho do SA relativamente á Solução Ótima para o problema de Job-Shop

De seguida serão apresentados os resultados para a instância FT06 de modo a entender melhor o funcionamento do *Simulated Annealing*.

4.1.1. Instância FT06

Relativamente a instância FT06, caracterizada por ser um problema do tipo *Job-Shop* em 6 tarefas alocadas a 6 máquinas. É possível verificar que foi possível encontrar a solução ótima para o mesmo e por este motivo ela será analisada com mais detalhe.

Pela análise do gráfico da evolução do *makespan* relativamente ao decorrer da temperatura depara-se que inicialmente este diminui rapidamente, mas depois estabiliza tornando-se muito mais moroso para atingir um *makespan* menor e assim minimizar a função objetivo.

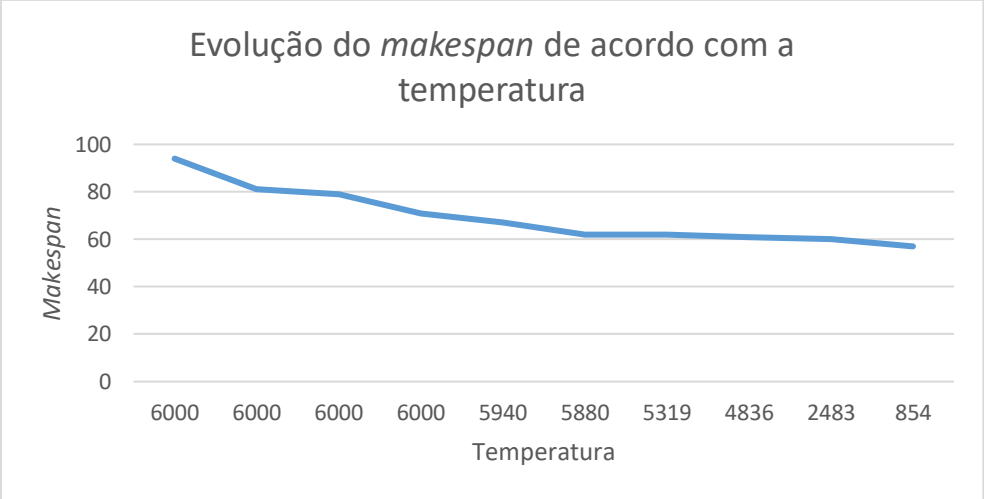


Gráfico 1 Evolução do *makespan* de acordo com a temperatura FT06

A figura 16 apresenta a entrada de dados da instância na ferramenta FT06.

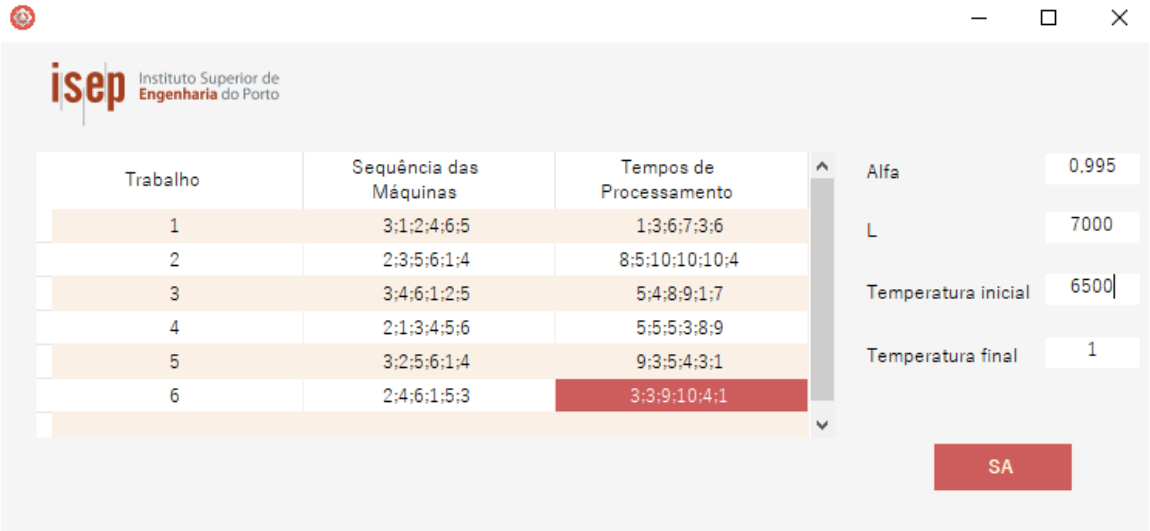


Figura 16 Introdução dos Dados no Protótipo da Instância FT06

O programa lógico segue uma estrutura de rede como mostra na figura 17. Cada círculo representa o trabalho e a máquina na sequência definida.

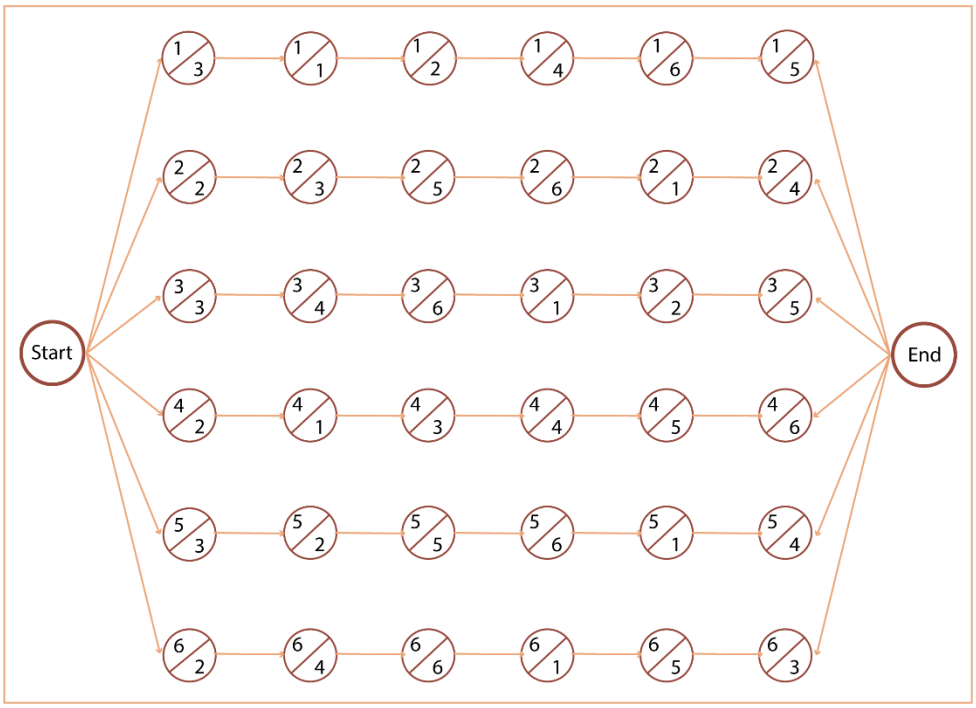


Figura 17 Estrutura de Rede da Instância FT06

O gráfico de *Gantt* para a solução ótima encontrada pode ser visualizado na figura 18.

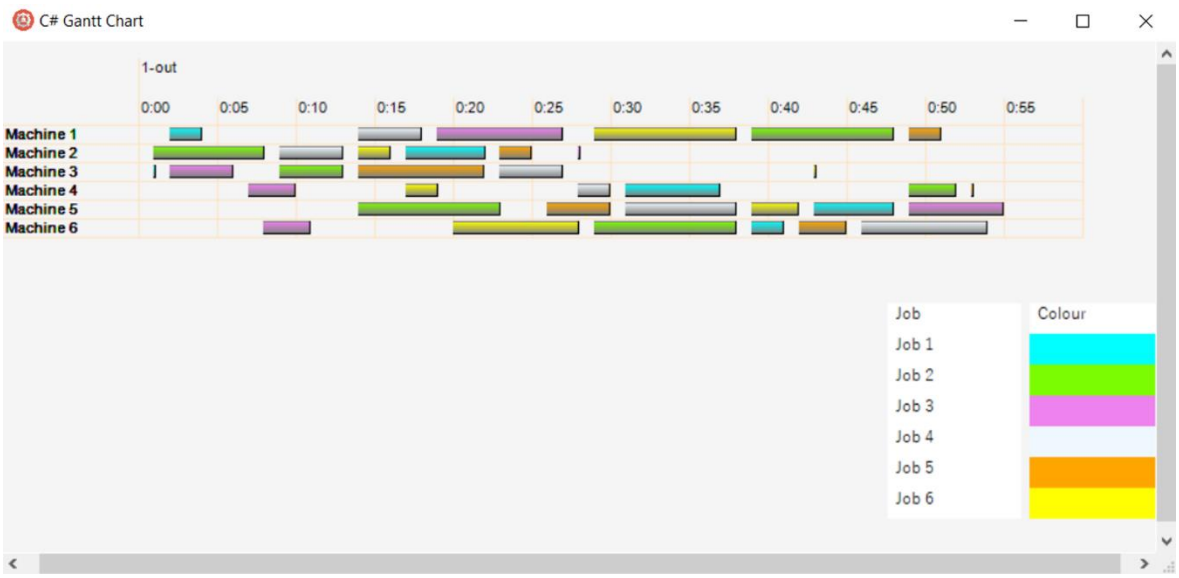


Figura 18 Gráfico de Gantt para a Instância FT06

4.2. Análise Estatística

4.2.1. Estatística Descritiva

Para o desenvolvimento da estatística descritiva os resultados foram analisados no programa IBM SPSS, sendo que as variáveis em estudo foram o *makespan*, o desvio absoluto da solução ótima e o desvio relativo da solução ótima.

A equação 16 representa o desvio absoluto da solução ótima, onde $f(x)_{SA}$ diz respeito ao desempenho do *Simulated Annealling* e $f(x)_{SO}$ o valor da solução ótima. O motivo da utilização desta medida de desempenho é por permitir comparar o desempenho mesmo em problemas distintos.

$$f(x)_{SA} - f(x)_{SO} \qquad 16$$

Assim, na figura 19 encontra-se representado um gráfico correspondente as instâncias corridas no protótipo para os problemas de escalonamento em *Flow-Shop* e na figura 20 os problemas do tipo J*ob-Shop.*

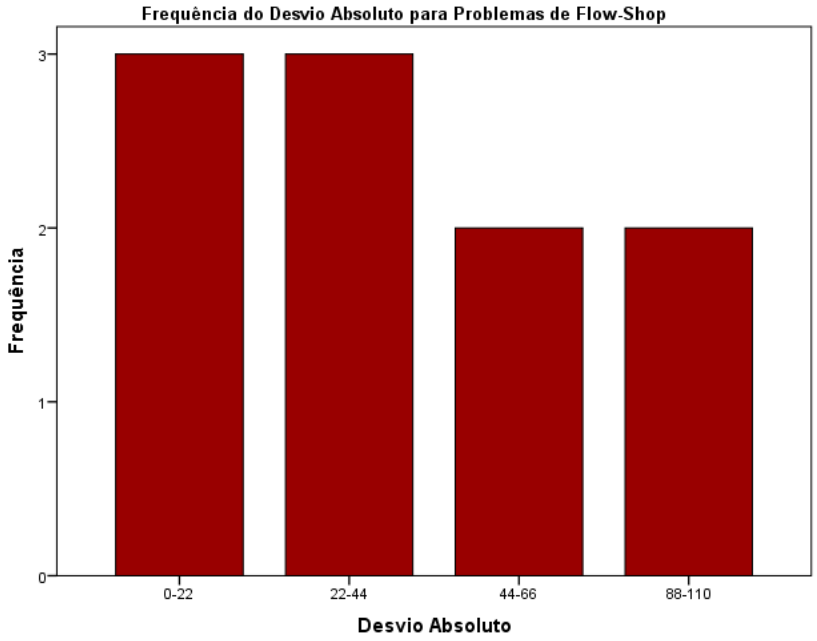


Figura 19 Gráfico de Barras do Desvio Absoluto das Instâncias para os problemas de *Flow-Shop*

Figura 20 Gráfico de Barras do Desvio Absoluto das Instâncias para os problemas de *Job-Shop*

Da ilação dos gráficos acima representados é possível identificar que o SA teve mais dificuldade em convergir na solução ótima para os problemas de J*ob-Shop*, mas apesar disso o desvio absoluto menor, entre 0 e 20, continuo a ser o com mais frequência entre as 10 instâncias corridas.

Para se poder inferir relativamente a mediana e os quartis dos dois tipos de problemas testados procederam-se a elaboração dos *boxplots* respetivos visíveis nas figuras 21 e 22.

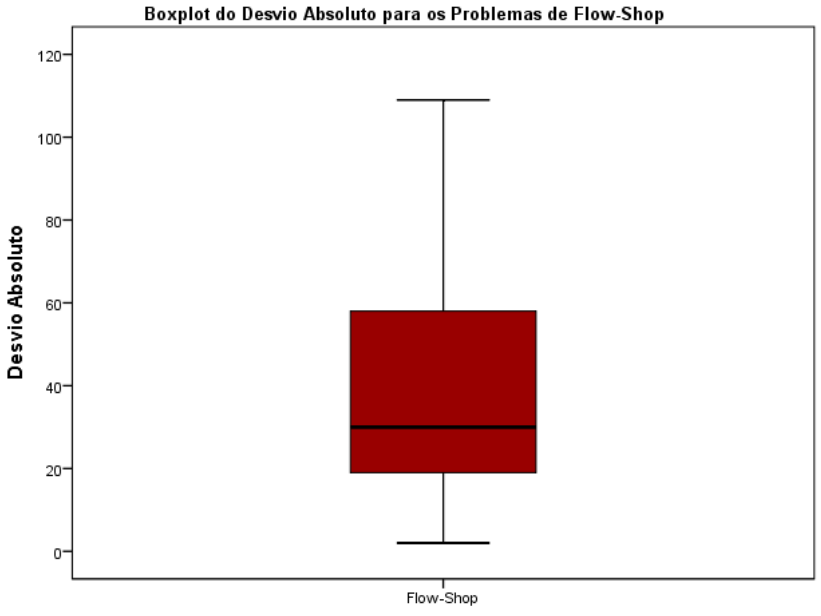


Figura 21 *Boxplot* do Desvio Absoluto das Instâncias para os problemas de *Flow-Shop*

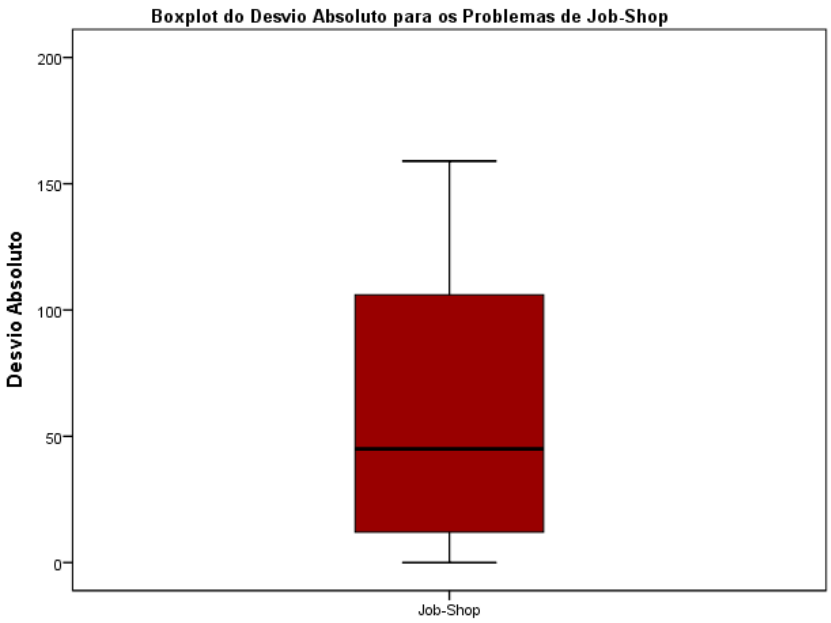


Figura 22 *Boxplot* do Desvio Absoluto das Instâncias para os problemas de *Job-shop*

Primeiramente denota-se que em nenhum dos casos ocorreu um *outlier* relativamente ao desvio absoluto. A mediana dos problemas de F*low-Shop* centrou-se em 30, enquanto os problemas de J*ob-Shop* obtiveram uma mediana de cerca de 45. A amplitude dos problemas de J*ob-Shop* é mais elevada indicando uma variação superior dos *makespan's* relativamente á solução ótima.

A média, mediana, desvio padrão e variância para os dois tipos de problemas testados poderão ser encontradas na tabela 8, onde é possível analisar que as instâncias corridas para os problemas em J*ob-Shop* tiveram um comportamento pior relativamente aos problemas em F*low-Shop* e estes últimos obtiveram menor dispersão dos resultados.

Tabela 8 Caracterização do Desvio Absoluto

	Flow Shop	*Job Shop*
Média	43,4	55,4
Mediana	30	45
Variância	1187,64	2521,24
Desvio Padrão	36,3263	52,9280

Caracteriza-se agora a estatística descritiva para o desvio relativo da solução ótima dado pela expressão apresentada na equação 17, sendo considerada a melhor métrica para comparar Meta-Heurísticas em múltiplas instâncias de um dado problema, pois considera o valor da solução ótima (Silberholz & Golden, 2010).

$$\frac{f(x)_{SA} - f(x)_{SO}}{f(x)_{SO}} \times 100 \qquad 17$$

O desvio da solução ótima, relativo, permite pondera os desvios das soluções relativamente as soluções ótimas demonstradas na literatura.

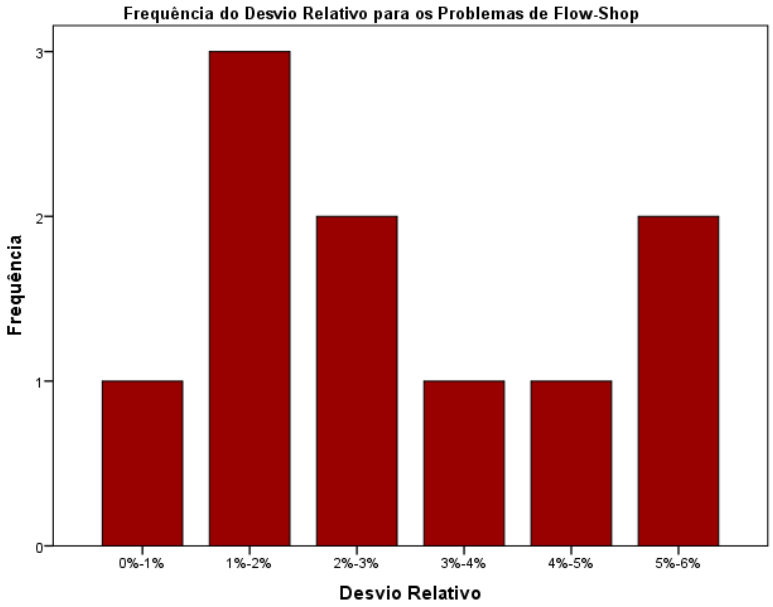


Figura 23 Desvio Relativo das Instâncias para os problemas de *Flow-Shop*

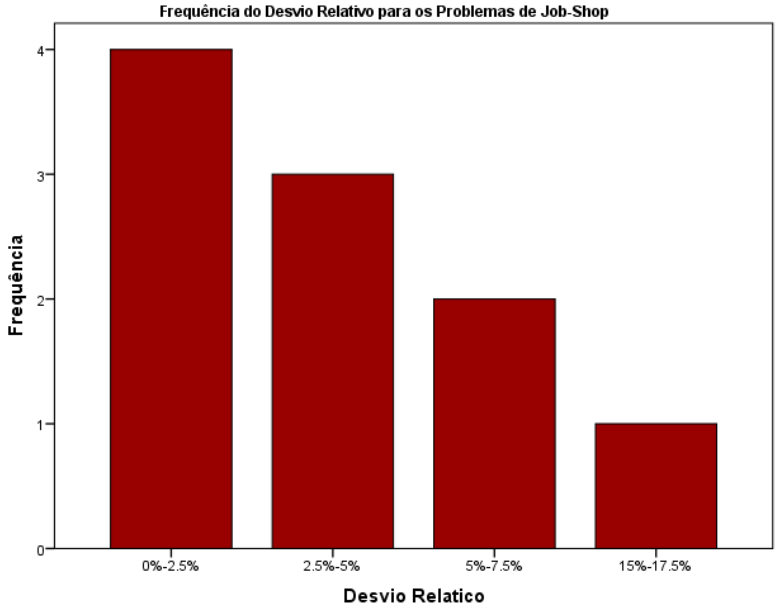


Figura 24 Desvio Relativo das Instâncias de para os problemas de *Job-Shop*

Os resultados do desvio relativo da solução ótima, do *Flow-Shop* e do *Job-Shop* podem ser analisados nas figuras acima representados, respetivamente. O *Flow-Shop* conseguiu obter resultados praticamente ótimos na maioria das instâncias. No entanto, o J*ob-Shop*, pode ser considerado mais disperso, apresenta soluções com desvios mais significativos.

O *boxplot* das instâncias do tipo de implementação industrial J*ob-Shop*, apresenta uma maior amplitude. Trata-se de uma amplitude significativamente superior à do F*low-Shop*, como pode ser analisado nas figuras abaixo representados (*Flow-Shop* e J*ob-Shop*, respetivamente).

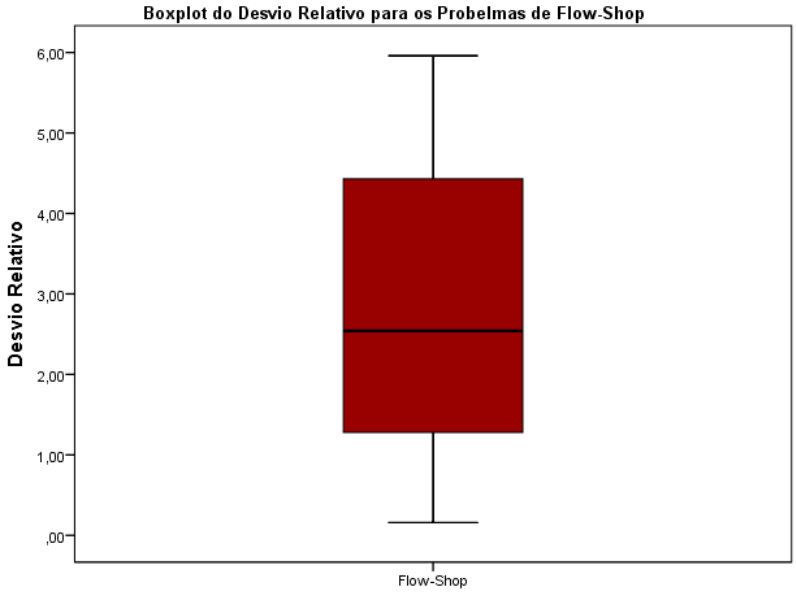


Figura 26 *Boxplot* do Desvio Relativo das Instâncias para os problemas de Flow-Shop

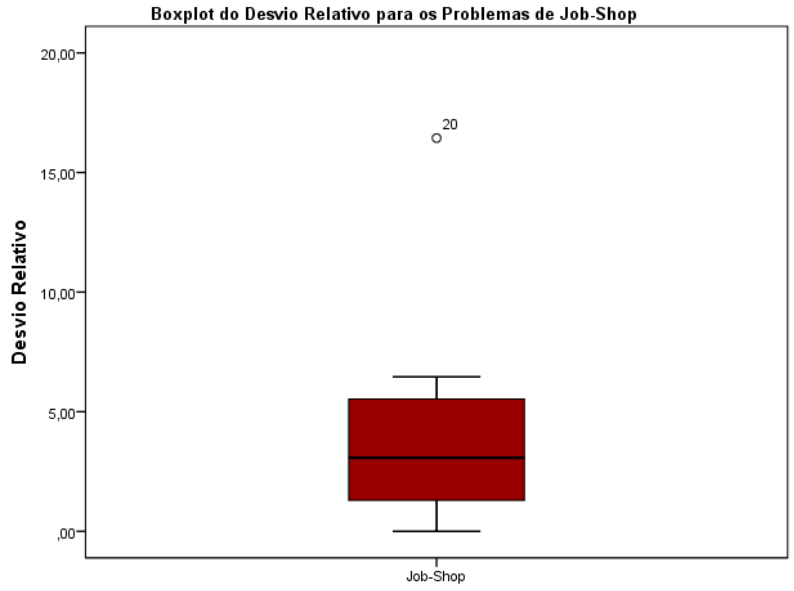


Figura 25 *Boxplot* do Desvio Relativo das Instâncias para os problemas de *Job-Shop*

É de denotar que nas instâncias do tipo de implementação industrial *Job-Shop* se obteve um *outlier* fixado nos 16,45% correspondente a instância YN1.

Na tabela 9 temos os parâmetros do desvio relativo, na qual podemos comparar os valores do *Flow-Shop* e do *Job-Shop*. Os resultados apresentados no *Flow-Shop* são, em todos os parâmetros, inferiores: com um desvio médio 2,86 no *Flow-Shop* e 4,23 no *Job-Shop*, logo têm uma diferença de 1,36. O *Flow-Shop* expõe um desvio padrão de 1,9568 e uma variância de 3,4463, por outro lado o *Job-Shop* tem um desvio padrão de 4,7437 e uma variância de 20,2529, ou seja, tem um desempenho adjacente do ótimo e praticamente constante.

Em suma, do estudo do *bloxplot* e dos parâmetros é possível concluir que existem certezas estatísticas do melhor desempenho das instâncias dos problemas em *Flow-Shop* no problema em estudo, quando analisado o desvio relativo.

Tabela 9 Desvio Relativo das Soluções

	Flow-Shop	*Job-Shop*
Média	2,86	4,23
Mediana	2,54	3,07
Variância	3,4463	20,2529
Desvio Padrão	1,9568	4,7437

4.2.2. Inferência Estatística

Para poder determinar se a meta-heurística teve um comportamento similar ao da literatura anteriormente identificada procedeu-se á realização dos testes de hipótese.

Tendo em conta o tamanho da amostra, isto é, a contabilização do número de instâncias, é inferior a 30 não é possível assumir a normalidade da amostra. Assim, para o desvio relativo das amostras efetuou-se o teste *Shapiro-Wilk* esquematizado na tabela 10 para os problemas de F*low-Shop* e na tabela 11 para os problemas dedicados de J*ob-Shop*.

Tabela 10 Teste de Normalidade para os problemas de *Flow-Shop*

Testes de Normalidade

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Estatística	df	Sig.	Estatística	df	Sig.
DesvioRelativo	,288	20	,000	,752	20	,000

Tabela 11Teste de Normalidade para os problemas de *Job-Shop*

Testes de Normalidade[a]

	Kolmogorov-Smirnov[b]			Shapiro-Wilk		
	Estatística	df	Sig.	Estatística	df	Sig.
DesvioRelativo	,274	10	,032	,727	10	,002

Os resultados demonstram que não se pode assumir que as instâncias analisadas seguem uma distribuição normal, pois *p-value* é de 0,000 para as instâncias de *Flow-Shop* e 0,002 para J*ob-Shop.* Sendo que ambas são menores do que 0,05, rejeitamos a normalidade.

Apesar da conclusão acima referida iremos realizar o teste T para amostras independentes de forma a concluir sobre o comportamento das soluções ótimas conseguidas. Os pressupostos do teste T para amostras independentes são:

1. As observações são normalmente distribuídas em cada população.

2. Homogeneidade de variâncias

3. Os dois grupos são amostras aleatórias independentes

As hipóteses são apresentadas nas equações 18 e 19, onde é considerado que μ_{SA} corresponde a distância média relativa do SA e μ_{SO} a distância média relativa dos problemas da literatura.

$$H0: \mu_{SA} - \mu_{SO} = 0 \quad 18$$

$$H1: \mu_{SA} - \mu_{SO} \neq 0 \quad 19$$

É possível visualizar os resultados do teste T nas seguintes tabelas:

Tabela 12 Teste T para as Instâncias para os problemas de *Flow-Shop*

	Teste de Levene para igualdade de variâncias		teste-t para Igualdade de Médias		teste-t para Igualdade de Médias		
	Z	Sig.	t	df	Sig. (2 extremidades)	Diferença média	Erro padrão de diferença
Variâncias iguais assumidas	7,907	,012	3,216	18	,005	4,59300	1,42823
Variâncias iguais não assumidas			3,216	9,000	,011	4,59300	1,42823

Tabela 13 Teste T para as Instâncias para os problemas de *Job-Shop*

	Teste de Levene para igualdade de variâncias		teste-t para Igualdade de Médias		teste-t para Igualdade de Médias		
	Z	Sig.	t	df	Sig. (2 extremidades)	Diferença média	Erro padrão de diferença
Variâncias iguais assumidas	20,691	,000	4,635	18	,000	2,86500	0,61812
Variâncias iguais não assumidas			4,635	9,000	,001	2,86500	0,61812

Em suma, não foi possível aceitar a hipótese nula para os dois tipos de problemas testados, pois o *p-value* de 0,000 permite assumir que há diferenças significativas no desempenho da meta-heurística testada e nos resultados das soluções ótimas apresentadas na literatura.

O teste de Mann-Whitney foi realizado seguidamente também para os dois tipos de problemas aqui estudados, assumindo a não normalidade provada dos dados. Assim no SPSS realizou-se o teste não paramétricos assumindo as hipóteses representadas nas equações 20 e 21.

$$H0: \eta_{SA} - \eta_{SO} = 0 \quad (20)$$

$$H1: \eta_{SA} - \eta_{SO} \neq 0 \quad (21)$$

Onde η_{SA} representa a mediana dos desvios relativos das soluções obtidas através da meta-heurística SA e η_{SO} diz respeito à mediana das soluções obtidas na literatura.

Nas tabelas 14 e 15 é possível visualizar os resultados obtidos:

Tabela 14 Teste não paramétrico para amostras independentes para os problemas de *Flow-Shop*

Estatísticas de teste[a]

	DesvioRelativo
U de Mann-Whitney	,000
Wilcoxon W	55,000
Z	-4,038
Significância Sig. (2 extremidades)	,000
Sig exata [2*(Sig. de 1 extremidade)]	,000[b]

Tabela 15 Teste não paramétrico para amostras independentes para os problemas de *Job-Shop*

Estatísticas de teste[a]

	DesvioRelativo
U de Mann-Whitney	,000
Wilcoxon W	55,000
Z	-4,038
Significância Sig. (2 extremidades)	,000
Sig exata [2*(Sig. de 1 extremidade)]	,000[b]

Indo de encontro ao teste t anteriormente realizado, é possível analisar que há diferenças significativas no desempenho da meta-heurística que se abordou ao longo desta dissertação. O p--value para as duas situações consideradas foi de 0,000 rejeitando a hipótese nula formulada.

5. CONCLUSÃO

Neste capítulo é elaborada uma análise crítica á dissertação desenvolvida, assim como a discriminação dos objetivos atingidos. Nessa sequência são ainda apresentadas as suas limitações e as propostas de investigações futuras.

5.1. Conclusões finais

Atualmente, há uma necessidade constante, por parte das organizações, em se moldar ao mercado que se mostra cada vez mais instável. Por consequência, as empresas são obrigadas a dar uma resposta rápida e eficaz. Toda esta instabilidade afeta a programação da produção que tem que se adaptar a todas as adversidades, tais como avarias e procura desnivelada.

Esta dissertação tem como objetivo colmatar as dificuldades sentidas por parte do escalonamento da produção, o que faz com que atividade se possa tornar demorosa e penosa. O protótipo desenvolvido visa a implementação de um sistema de escalonamento adaptável a qualquer sistema de produção.

A meta-heurística implementada no protótipo foi o *Simulated Annealling*, que em oposição a métodos determinísticos que têm a desvantagem de ficarem presos a mínimos locais, a principal vantagem deste algoritmo é a sua capacidade de evitar ficar preso em ótimos locais, foi provado que irá convergir para o ótimo global se for usado aleatoriamente combinado com um arrefecimento muito lento. A medida de desempenho executada foi a minimização do *makespan*, simbolizada por instante final de processamento de todas as ordens de produção em todas as máquinas.

O protótipo, sendo este *userfriendly*, permite a fácil introdução dos dados por parte do utilizador, este apenas tem de ter conhecimento dos parâmetros da meta-heurística que está implementada. Foi concretizado num sistema modular que possibilita a junção de novas meta-heurísticas e novas medidas de desempenho. As interfaces de apresentação das soluções são de fácil compreensão, sendo possível apresentar a solução num gráfico de *Gantt*. Gostava ainda de salientar que o protótipo consente todos os tipos de implementação industrial incluindo o *Job-Shop*.

Com vista a estudar o desempenho e a eficácia da meta-heurística implementada, foram realizados dez testes computacionais em tipos de problemas F*low-Shop* e outros dez em *Job-Shop*. As experiências foram realizadas com base numa seleção de problemas padrão conhecidos na literatura, caracterizados por diferentes níveis de tamanho e dificuldade.

Foram obtidos melhores resultados para as instâncias de *Flow-Shop*, que é compreensível pois os problemas de planeamento do tipo *Job-Shop* são de complexidade mais elevada. Os testes de hipótese paramétricos e não paramétricos, permitiram concluir um desempenho não satisfatório da meta-heurística implementada, rejeitando todas as hipóteses nulas consideradas.

Em suma, foi concretizado o principal objetivo traçado definido por a implementação de uma ferramenta ágil, de fácil utilização e que permite uma visualização gráfica das soluções, nomeadamente, através de um gráfico de *Gantt*. O protótipo tem ainda como principal característica a permissão de resolução de qualquer tipo de problema de escalonamento, até mesmo do tipo J*ob-Shop*.

5.2. Limitações e investigação futura

A maior adversidade sentida na implementação da meta-heurística SA, foi o facto de para convergir para a solução ótima, ser necessário uma minuciosa escolha dos parâmetros. Assim para ultrapassar esta limitação, proponho o desenvolvimento e implementação de uma meta-heurística diferente que obtenha, também, melhores resultados.

Ainda como trabalho futuro a ser desenvolvido, proponho o posterior aumento do número de instâncias consideradas para cada tipo de problema de escalonamento e a extensão de novas medidas de desempenho que com este crescimento da investigação torna possível uma análise de comportamento ainda mais generalizada.

Como último ponto de reflexão, propõe-se que numa investigação futura se inclua no protótipo os tempos de preparação das máquinas, e até mesmo escalonamentos dinâmicos, considerando adversidades como por exemplo avarias.

REFERÊNCIAS BIBLIOGRÁFICAS

Adams, J., Balas, E., & Zawack, D. (1988). The shifting bottleneck procedure for job shop scheduling. *Management Science*, *34*, 391–401.

Adams, Joseph, Bals, E., & Zawack, D. (1988). The Shifting Bottleneck Procedure for Job Shop Scheduling. *Management Science*, *34*(3).

Agin. (1996). Optimum seeking with branch and bound. *Management Science*, *13*.

André Santos. (2020). *Auto-Parametrização de Meta-Heurísticas para Problemas de Escalonamento em Ambiente Industrial*.

Anily, S., & Federgruen, A. (1987). Simulated Annealing Methods with General Acceptance Probabilities. *Journal of Applied Probability*.

Baker, K. R. (1974). *Introdução ao sequenciamneto e programaçao*. Retrieved from https://openlibrary.org/books/OL5047415M/Introduction_to_sequencing_and_scheduling

Baker, K. R., & Trietsch, D. (2009). *Principles of Sequencing and Scheduling*. https://doi.org/10.1002/9780470451793

Binh Ho, N., & Cing Tay, J. (2018). Solving Multiple-Objective Flexible Job Shop Problems by Evolution and Local Search. *IEEE Access*, *38*(5), 674–685. Retrieved from https://ieeexplore.ieee.org/document/4603098

Blazewicz, J., Weglarz, J., Machowiak, M., Kovalyov, M., & Trystram, D. (2004). Escalonamento de tarfeas em processadores paralelos para minimizar o tempo de execução. *Annals of Opperations Research*.

Blazewicz, Jacek, Domschke, W., & Pesch, E. (1996). O problema do escalonamento em Job Shop: técnicas de solução convencionais e novas. *European Journal of Operational Research*, 1–33. Retrieved from https://www.sciencedirect.com/sdfe/pdf/download/eid/1-s2.0-0377221795003622/first-page-pdf

Blazewicz, Jacek, Ecker, K. H., Pesch, E., Schmidt, G., & Weglarz, J. (2007). Handbook on Scheduling. In *Handbook on Scheduling*. https://doi.org/10.1007/978-3-540-32220-7

Blum, C., & Roli, A. (2003). Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys*, 268–308. Retrieved from https://www.researchgate.net/profile/Christian_Blum/publication/221900771_Metaheuristics_in_Combinatorial_Optimization_Overview_and_Conceptual_Comparison/links/02bfe510587dd1ca65000000.pdf

Brandimarte, P. (1993). Roteamento e programação em Job Shop por pesquisa de tabu. *Annals of Operations Research*.

Brucker, P. (2006). *Scheduling Algoritms*. https://doi.org/10.1017/CBO9781107415324.004

Cho, H.-S., Paik, C.-H., Yoon, H.-M., & Kim, H.-G. (2005). A Robust Design of Simulated Annealing Approach for Mixed-Model Sequencing. *Computer & Industrial Engineering*.

Coffman, E. G., & R. L. Graham. (1971). *Optimal Scheduling for Two-Processor Systems*. https://doi.org/10.1007/BF00288685

Conway, R. W., Maxwell, W. L., & Miller, L. W. (1967). *Teoria do Escalonamento*. Addison-Wesley Publishing Company.

Cruz, H. R. (2018). *Planeamento e Programação da Produção- Sistemas Integrados de Gestão de Informação na base da Tomada de Decisão*.

Cui, M., Bai, R., Lu, Z., Li, X., Aickelin, U., & Ge, P. (2019). Regular Expression Based Medical Text Classification Using Constructive Heuristic Approach. *IEEE Access*.

D. Applegate, W. C. (1991). A computational study of the job-shop scheduling instance. *ORSA Journal on Computing*, *3*, 149–156.

Eglese, R. W. (1990). Simulated Annealing: A Tool for Operational Research. *European Journal of Operational Research*.

El-Ghazali, T. (2009). *MetaHeuristics:from design to implementation*. https://doi.org/10.16309/j.cnki.issn.1007-1776.2003.03.004

Engelbrecht, A. P. (2007). *Computational Intelligence: An Introduction*.

Fang, H., Ross, P., & Corne, D. (1993). *Uma abordagem promissora do algoritmo genético para problemas de escalonamento, reprogramação e Open Shop*.

Figueiredo, J. (2015). *Implementação de um Algoritmo Genético Híbrido com Simulated Annealing para o problema Job Shop*. Retrieved from https://repositorium.sdum.uminho.pt/bitstream/1822/34374/1/José Filipe Moreira da Silva Figueiredo.pdf

Fisher, H., & Thompson, G. L. (1963). Probabilistic learning combinations of local job-shop scheduling rules. *Industrial Scheduling*, 225–251.

Fleischer, M., & Jacobson, S. H. (1999). Information Theory and the Finite-Time Behavior Simulated Anealing Algoritm: Experimental Results. *Journal of Computing*.

French, S. (1982). *Sequencing and Scheduling: An introduction to the Mathematics of the Job Shop*. Chichester: Ellis Horwood.

Gendreau, M., & Potvin, J. (2010). *Handbook of Metaheristics*. Springer.

Gil, A. C. (2008). *Métodos e técnicas de pesquisa soial* (6ª). São Paulo: Atlas.

Glover, F., & Kochenberger, G. A. (2003). *Handbook of Metaheuristics*. Springer.

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Kan, A. H. G. R. (1979). Optimization and approximation in deterministic sequencing and scheduling theory: a survey. *Annals of Discrete MAthematics*.

Hansen, P., & Mladenovic, M. (1999). An introduction to variable neighborhood search. *Advances and Trends in Local Search Paradigms for Optimization*.

Hedar, A., & Fukushima, M. (2002). Hybrid simulated annealing and direct search method for nonlinear unconstrained global optimization. *Optimization Methods and Software*.

Hedar, A., & Fukushima, M. (2006). Meta-heuristics programming. *Proceedings of 2nd International Workshop on Computational Intelligence Applications*.

Hedar, A., Mabrouk, E., & Fukushima, M. (2011). Tabu programming: A problem solver through adapive memory programming over tree data structuer. *International Journal of Information Technology Decision Making*, 373–406.

Jiang, T., & Zhang, C. (2018). Application of Grey Wolf Optimization for Solving Combinatorial Problems_ Job Shop and Flexible Job Shop Scheduling Cases. *IEEE Journals & Magazine*, 26231–26240. Retrieved from https://ieeexplore.ieee.org/abstract/document/8355479

Kim, D.-W., Kim, K.-H., Jang, W., & Chen, F. F. (2002). Unrelated parallel machine scheduling with setup times using simulated annealing. *Robotics and Computer Integrated Manufacturing*, 223–231.

Kirkpatric, S., Gelatt Jr., C. D., & Vecci, M. P. (1983). Optimization by Simulated Annealing. *Science*,

671–680.

Koskinen, J., Raduly-Baka, C., Johnsson, M., & Nevalainen, O. S. (2020). Rolling horizon production scheduling of multi-model PCBs for several assembly lines. *International Journal of Production Research*, *58*(4), 1052–1073. https://doi.org/10.1080/00207543.2019.1609708

Laarhoven, P. J. M. van, & Arts, E. H. L. (1987). Simulated Annealing: Theory and Applications. *Mathematics and Its Applications*.

Lawrence, S. (1984). *Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques*.

Lopez, P., & Roubellat, F. (2008). *Production Scheduling*. UK, USA: ISTE and Wiley.

Loureiro, N. F. P. (2014). *Utilização do Simulated Annealing na resolução de problemas no planeameto de produção*.

Low, C. (2005). Simulated annealing heuristic for flow shop scheduling problems with unrelated parallel machines. *Computers & Operations Research*. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054804000048?via%3Dihub

Macedo, P., Zacarias, M. S., & Tribolet, J. (2002). *Técnicas e Métodos de Investigação em Engenharia Organizacional : Projeto de Investigação em Modelação de Processos de Produção*.

Madureira, A. (2003). *Aplicação de Meta-Heurísticas ao Problema de Escalonamento em Ambiente Dinâmico de Produção Discreta*. Universidade do Minho.

Madureira, A. (2009). *Técnicas Emergentes de Optimização no Suporte à Tomada de Decisão*.

Marques, J. A. de S. (2015). *Heurísticas de pesquisa local para problemas de máquina única*.

Martins, R. A., & Sacomano, J. B. (1994). Integração, Flexibilidade e Qualidade: os caminhos para um novo paradigma produtivo. *Gestão & Produção*, *1*(2), 153–170. Retrieved from http://www.scielo.br/pdf/gp/v1n2/a03v1n2.pdf

Mattfeld, D. C., & Vaessens, R. J. M. (1992). instances. Retrieved from http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/jobshop1.txt

Morton, T., & Pentico, D. W. (1993). *Sistemas de agendamento heurístico: com aplicações em sistemas de produção e gestão de projetos*. Wiley.

Motaghedi-larijani, A., Sabri-laghaie, K., & Heydari, M. (2010). Solving flexible Job Shop scheduling with multi objective approach. *International Journal of Industrial Engineering & Production Research*, 197–209.

Nouiri, M., Bekrar, A., Jemai, A., Niar, S., & Ammari, A. C. (2018). An effective and distributed particle swarm optimization algorithm for flexible job-shop scheduling problem. *Journal of Intelligent Manufacturing*, 603–615. Retrieved from https://link.springer.com/article/10.1007/s10845-015-1039-3

Osman, M. (2011). *Designing Machine Learning Tools Based on Meta-Heuristic Programming*. Retrieved from https://www.researchgate.net/publication/278404748_Designing_Machine_Learning_Tools_Based_on_Meta-Heuristic_Programming

Pacheco, R. F., & Santoro, M. C. (1999). Proposta de classificação hierarquizada dos modelos de solução para o problema de Job Shop scheduling. *Gestão & Produção*.

Papadimitrou, C. H., & Kenneth, S. (1998). *Combinatorial Optimization: Algortithms and Complexity*. Toronto.

Park, M.-W., & Kim, Y.-Da. (1998). A Systematic Procedure for Setting Parameters in Simulated Annealing Algorithms. *Computers & Operations Research*.

Park, M. W., & Kim, Y. D. (1998). A Systematic procedure for Setting Parameters in Simulated Annealing Algoritms. *Computers & Operations Research*.

Pentico, D. W., & Morton, T. E. (1993). *Heuristic Scheduling System*. Wiley.

Pérez, M. A. F., & Raupp, F. M. P. (2016). A Newton-based heuristic algorithm for multi-objective flexible job-shop scheduling problem. *Journal of Intelligent Manufacturing*, 409–416. https://doi.org/10.1007/s10845-014-0872-0

Pezzella, F., Morganti, G., & Ciaschetti, G. (2008). *A genetic algorithm for the Flexible Job-shop Scheduling Problem*. *35*, 3202–3212. https://doi.org/10.1016/j.cor.2007.02.014

Pinedo, M. L. (2008). *Scheduling: Theory, Algorihms and Systems*. New York: Springer.

Rose, J., Klebsch, W., & Wolf, J. (1990). Temperature Measurement and Equilibrium Dynamics of Simulated Annealing Placement. *IEEE Transactions on Computer Aided Design*.

Rose, Jonathan, Klebsch, W., & Wolf, J. (1990). Temperature Measurement and Equilibrium Dynamics of Simulated Annealing Placements. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*.

Roshanzamir, M., Palhang, M., & Mirzaei, A. (2019). Graph structure optimization of Genetic Network Programming with ant colony mechanism in deterministic and stochastic environments. *Swarm and Evolutionry Computation*, *51*. Retrieved from https://www.sciencedirect.com/science/article/pii/S221065021831068X

Santos, A. (2015). *Análise do Desempenho de Técnicas de Otimização*.

Sarangi, A., Samal, S., & Sarangi, S. K. (2019). Comparative Analysis of Cauchy Mutation and Gaussian Mutation in Crazy PSO. *International Conference on Computing and Communications Technologies*. Retrieved from https://ieeexplore.ieee.org/document/8824972/authors#authors

Shahzad, Atif; Mebarki, N. (2016). Learning dispatching rules for scheduling: A synergistic view comprising decision trees, Tabu search and simulation. *Computers*. Retrieved from https://doi.org/10.3390/computers5010003

Silberholz, J., & Golden, B. (2010). Comparison of Metaheuristics. Handbook of etaheuristics. In *Internacional Series in Operation Research & Management Sciences*. Springer.

Smith, M. (1986). Characteristics of U.S. Flexible Manufacturing Systems - A Survey. *Proceedings of the Second ORSA/TIMS Conference on Flexible Manufacturing Systems: Operations Research Models and Applications*. Amesterdam: Elsevier Science Publishers B.V.

Storer, R. H., Wu, S. D., & Vaccari, R. (1992). New search spaces for sequencing instances with application to job shop scheduling. *Management Science*, *38*, 1495–1509.

Sule, D. (1997). *Industrial Scheduling*. Boston: PWS publishing Company.

Taillard, E. (1989). *Benchmarks For Basic Scheduling Problems*. 1–17.

Taillard, E. (1993). Benchmarks for basic scheduling problems. Retrieved from http://mistic.heig-vd.ch/taillard/problemes.dir/ordonnancement.dir/ordonnancement.html

Tavares, H. M. G. (2015). *Estudo e Análise do Sequenciamento de Tarefas de Produção : Job Shop Scheduling*.

Varela, M. L. R. (2007). *Uma Contribuição para o Escalonamento da Produção baseado em Métodos Globalmente Distribuídos* (Universidade do Minho). Retrieved from

https://core.ac.uk/download/pdf/55608244.pdf

Wang, Ping; Sang, Hongyan; Guo, Hengwei; Li, J. (2020). Improved Migrating Birds Optimization Algorithm to Solve Hybrid Flowshop Scheduling Problem With Lost-Streaming. *IEEE Access*, *8*. Retrieved from https://ieeexplore.ieee.org/document/9091160/authors#authors

Wu, A. S., & Banzhaf, W. (1998). Variable-lenght representation and noncoding segments for evolutionary algorithms. *Evolutionary Computation*.

Yamada, T., & Nakano, R. (1992). A genetic algorithm applicable to large-scale job-shop instances. *Parallel Instance Solving from Nature*, *2*, 281–290.

Yang, X.-S. (2010a). A New Metaheuristic Bat-Inspired Algorithm. *Studies in Computational Intelligence*, *284*. Retrieved from https://link.springer.com/chapter/10.1007/978-3-642-12538-6_6#citeas

Yang, X.-S. (2010b). *Nature-inspired Metaheuristic Algorithms* (L. Press, Ed.). Retrieved from https://www.researchgate.net/profile/Xin-She_Yang/publication/235979455_Nature-Inspired_Metaheuristic_Algorithms/links/0c96051520ff6131a0000000/Nature-Inspired-Metaheuristic-Algorithms.pdf

Zhuang, Z., Huang, Z., Chen, L., & Qin, A. W. (2019). Uma Heuristica baseadana rede complexa de problemas de escalonamento de Open Shop com tempos de Setp e prazos de entrega dependesntes da sequência. *IEEE Access*, *7*, 140946–140956. https://doi.org/10.1109/ACCESS.2019.2944296