FACULDADE DE CIÊNCIAS E TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA MECÂNICA E INDUSTRIAL

JOSÉ ANTÓNIO PEREIRA GONÇALVES

Mestrado/Licenciatura em Nome do Grau Anterior

APLICAÇÃO DE AGENDAMENTO SOBRE EXAMES PET/CT

ALGUMAS CONSIDERAÇÕES SOBRE A VIDA, O UNIVERSO E TUDO O MAIS

MESTRADO EM ENGENHARIA E GESTÃO INDUSTRIAL

Universidade NOVA de Lisboa

Draft: *16 de agosto de 2025*

FACULDADE DE
CIÊNCIAS E TECNOLOGIA

DEPARTAMENTO DE
ENGENHARIA MECÂNICA E INDUSTRIAL

APLICAÇÃO DE AGENDAMENTO SOBRE EXAMES PET/CT

ALGUMAS CONSIDERAÇÕES SOBRE A VIDA, O UNIVERSO E TUDO O MAIS

JOSÉ ANTÓNIO PEREIRA GONÇALVES

Mestrado/Licenciatura em Nome do Grau Anterior

Orientadora: Mary Doe Adviser Name
Full Professor, NOVA University Lisbon

Coorientadores: John Doe Co-Adviser Name
Associate Professor, NOVA University Lisbon

John Doe other Co-Adviser Name
Full Professor, NOVA University Lisbon

MESTRADO EM ENGENHARIA E GESTÃO INDUSTRIAL

Universidade NOVA de Lisboa

Draft: *16 de agosto de 2025*

Resumo

Independentemente da língua em que a dissertação está escrita, geralmente esta contém pelo menos dois resumos: um resumo na mesma língua do texto principal e outro resumo numa outra língua.

A ordem dos resumos varia de acordo com a escola. Se a sua escola tiver regulamentos específicos sobre a ordem dos resumos, o template (LaTeX) NOVAthesis LaTeX (novathesis) irá respeitá-los. Caso contrário, a regra padrão no template novathesis é ter em primeiro lugar o resumo *no mesmo idioma do texto principal* e depois o resumo *no outro idioma*. Por exemplo, se a dissertação for escrita em português, a ordem dos resumos será primeiro o português e depois o inglês, seguido do texto principal em português. Se a dissertação for escrita em inglês, a ordem dos resumos será primeiro em inglês e depois em português, seguida do texto principal em inglês. No entanto, esse pedido pode ser personalizado adicionando um dos seguintes ao arquivo `5_packages.tex`.

```
\abstractorder(<MAIN_LANG>):={<LANG_1>,...,<LANG_N>}
```

Por exemplo, para um documento escrito em Alemão com resumos em Alemão, Inglês e Italiano (por esta ordem), pode usar-se:

```
\ntsetup{abstractorder={de={de,en,it}}}
```

Relativamente ao seu conteúdo, os resumos não devem ultrapassar uma página e frequentemente tentam responder às seguintes questões (é imprescindível a adaptação às práticas habituais da sua área científica):

1. Qual é o problema?
2. Porque é que é um problema interessante/desafiante?
3. Qual é a proposta de abordagem/solução?
4. Quais são as consequências/resultados da solução proposta?

Palavras-chave: Primeira palavra-chave, Outra palavra-chave, Mais uma palavra-chave, A última palavra-chave

Abstract

Regardless of the language in which the dissertation is written, usually there are at least two abstracts: one abstract in the same language as the main text, and another abstract in some other language.

The abstracts' order varies with the school. If your school has specific regulations concerning the abstracts' order, the novathesis (LaTeX) template will respect them. Otherwise, the default rule in the novathesis template is to have in first place the abstract in *the same language as main text*, and then the abstract in *the other language*. For example, if the dissertation is written in Portuguese, the abstracts' order will be first Portuguese and then English, followed by the main text in Portuguese. If the dissertation is written in English, the abstracts' order will be first English and then Portuguese, followed by the main text in English. However, this order can be customized by adding one of the following to the file `5_packages.tex`.

```
\ntsetup{abstractorder={<LANG_1>,...,<LANG_N>}}
\ntsetup{abstractorder={<MAIN_LANG>={<LANG_1>,...,<LANG_N>}}}
```

For example, for a main document written in German with abstracts written in German, English and Italian (by this order) use:

```
\ntsetup{abstractorder={de={de,en,it}}}
```

Concerning its contents, the abstracts should not exceed one page and may answer the following questions (it is essential to adapt to the usual practices of your scientific area):

1. What is the problem?
2. Why is this problem interesting/challenging?
3. What is the proposed approach/solution/contribution?
4. What results (implications/consequences) from the solution?

Keywords: One keyword, Another keyword, Yet another keyword, One keyword more, The last keyword

Índice

Índice de Figuras **iv**

List of Algorithms **v**

Siglas **vi**

1 Introdução **1**
1.1 Motivação e Contexto 1
1.2 Apresentação do Problema e Objetivo 1
1.3 Abordagem Metodológica 2

2 Base Teórica **3**
2.1 Extensões, Restrições, e Critérios 3
2.2 Abordagens para a Resolução 4
2.2.1 Algoritmos Exatos 4
2.2.2 Heurísticas 5
2.2.3 Meta-Heurísticas 5

3 Revisão de Literatura **7**
3.1 Abordagens para a Resolução 7
3.1.1 Algoritmos Exatos 7
3.1.2 Heurísticas 8
3.1.3 Meta-Heurísticas 8

4 Mapeamento dos Exames **10**

5 Desenvolvimento dos Modelos **12**
5.1 Problema de *Makespan* 12
5.2 Problema do Número de Exames 20

Bibliografia **25**

Índice de Figuras

5.1 Codificação do modelo 1 15
5.2 Codificação do modelo 2 16
5.3 Impacto na qualidade e tempo computacional das combinação dos níveis para o Modelo 1 NGV 17
5.4 Impacto na qualidade e tempo computacional das combinação dos níveis para o Modelo 1 GV 18
5.5 Impacto na qualidade e tempo computacional das combinação dos níveis para o Modelo 1 GV 19
5.6 Codificação do modelo 1 23
5.7 Codificação do modelo 2 23
5.8 Codificação do modelo 3 23

List of Algorithms

SIGLAS

novathesis NOVAthesis LaTeX *(pp. i, ii)*

1

Introdução

Este capítulo irá introduzir o tema desta dissertação e a sua estrutura, a motivação e contexto, os objetivos a atingir e a abordagem tomada.

1.1 Motivação e Contexto

O custo associado ao setor de saúde tem vindo a aumentar, em que o grupo dos países do G7 a pagar mais de 10% do seu GDP para este fim. O elevado custo deve-se a vários fatores, ao envelhecimento da população, à degradação do estilo de vida, ao aumento da estadia em hospital, entre outros [1].

A realização de exames é um das principais atividades de um hospital, como forma de apoio ao diagnóstico e à cirurgia. Em Portugal existe por isso tempos máximos de resposta garantidos [2], medicina nuclear (MN) insere-se no meio complementar de diagnóstico e terapêutica, devendo ocorrer um exame desta natureza em menos de 30 dias a partir da indicação clínica. Contudo, frequentemente estes limites não são compridos, devido à maior procura destes meios [3] e a ineficiências na realização desta atividade.

Lean Healthcare apresenta-se como uma das solução para combater as ineficiências presentes no setor de saúde, para tal existe uma grande variedade de abordagens mas também muitos desafios na implementação de soluções.

1.2 Apresentação do Problema e Objetivo

Esta dissertação teve como *pich* a melhoria da eficiência do departamento de MN do Hospital Garcia de Orta que, a priori, seria através da melhoria do agendamento de pacientes ao longo do dia de trabalho de forma a realizar um número maior de exames com os recursos humanos e físicos.
Cada exame é composto por várias tarefas, na qual podem ser necessários vários recursos

diferentes, não ocorrendo tempo de espera entre tarefas adjacentes de um mesmo exame. Esta dissertação pretende alcançar três objetivos:

1. O mapeamento dos exames mais frequentemente realizados, com possibilidade de mapear os exames que ocorrem com menor frequência;
2. A criação de ferramentas de apoio à decisão, duas em contexto operacional e outra para contexto tática;
3. A validação de resultados obtidos pelas ferramentas em termos de viabilidade e robustez.

1.3 Abordagem Metodológica

De forma a atingir os objetivos previamente referidos é necessário estruturar a abordagem metodológica. Será realizada uma revisão de literatura que abordará os temas de mapeamento, tipos de problemas de agendamento e respetivos métodos de resolução, e **outra coisa?**. De seguida serão mapeados e validados os exames, e posteriormente recolhidos dados relativos à duração de cada atividade. Seguidamente serão desenvolvidos os modelos de apoio à decisão já mencionados, com aplicação destes no sistema real com apresentação dos resultados obtidos. Finalmente serão propostos trabalhos futuros, e serão retiradas conclusões.

2

Base Teórica

Este capítulo irá introduzir alguns conhecimentos necessários para comunicar eficientemente a revisão da literatura e o trabalho realizado de forma geral.

2.1 Extensões, Restrições, e Critérios

O problema aqui apresentado poderá ser englobado na família de problemas *Job-Shop*, um dos problemas mais estudados na área de investigação operacional.

Este problema é definido por um conjunto finito J de trabalhos, um conjunto finito R de recursos, e um conjunto finito O de operações. Cada trabalho j é uma sequência de n_j operações consecutivas em O, em que cada operação $i \in O$ será processada no recurso R_i com duração contínua de τ_i e sem antecipação de operações [4].

Contudo, dificilmente o problema em estudo será descrito pela definição clássica de *Job-Shop*, existem então uma grande variedade de extensões, restrições, e critérios. Permitindo modificar o problema base de forma a representar de forma realista o sistema em estudo. Desta forma, o problema em estudo poderá ser descrito como *Flexible Multi-Resource Job Shop with No-Wait*.

A extensão *Flexible* foi descrito pela primeira vez em [5] e dado o nome de *Flexible* em [6], esta adição é uma das mais importantes na generalização do problema *Job-Shop* para o contexto real, existe no entanto o aumento da complexidade e a impossibilidade de utilizar o método tradicional na sua resolução [4]. Existe, para cada operação $i \in O$ um conjunto de recursos $R_i \subseteq R$ que podem ser utilizados, ou seja, existe um conjunto de recursos-tipo que exercem a mesma função, com duração de τ_i^k para o recurso $k \in R_i$,

A extensão *Multi-Resource* foi descrito pela primeira vez em [7], [8] na qual existe, para cada operação $i \in O$, $m(i)$ recursos diferentes, tal que o conjunto R_i^k contem o k

recurso-tipo da operação O_i, ou seja, cada operação será processada por $m(i)$ recursos distintos.

A restrição *Minimum/Maximum Time Lag* limita o tempo decorrido entre operações sequenciais de um trabalho, sendo *No-Wait* o caso extremo desta restrição, garantindo que não existe tempo de espera entre operações sequenciais, este conceito foi proposto em [9], tal que, o começo da operação O_{i+1} ocorra no mesmo momento que a operação O_i termina, caso estas façam parte do mesmo trabalho j.

Ao mesmo tempo também é possível resolver o problema apresentado de forma a maximizar ou minimizar diferentes critérios. Estes podem ser baseado em tempo, número de trabalhos, custo, receita, ou impacto no meio ambiente.

2.2 Abordagens para a Resolução

Existem três categorias gerais para a resolução de problemas *Job-Shop*, algoritmos exatos, heurísticas, e meta-heurísticas [10]. De seguida serão apresentados alguns métodos de resolução do problema *Job-shop* e uma breve descrição de como funcionam.

2.2.1 Algoritmos Exatos

Historicamente, a utilização de métodos exatos foi a principal abordagem utilizada, com especial ênfase nos artigos iniciais sobre o problema clássico de *Job-Shop*. A utilização de *Integer Linear Programming*(*ILP*), *Mixed Integer Linear Programming* (*MILP*) são as principais abordagens utilizadas.
A forma canónica de *Linear Programming* pode ser expressa por:

$$\text{Maximizar } z = c_1x_1 + c_2x_2 + \ldots + c_nx_n$$

Sujeito a:

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1$$

$$a_{21}x_2 + a_{22}x_2 + \ldots + a_{2n}x_n \leq b_2$$

$$\vdots$$

$$a_{m1}x_m + a_{m2}x_2 + \ldots + a_{mn}x_n \leq b_m$$

$$x_j \geq 0, j = 1, \ldots, n$$

Existem várias modelos utilizados para a representar o sistema em estudo, o modelo disjuntivo, o modelo indexado no tempo, e o modelo baseado em ordenação.

O modelo disjuntivo utiliza um grafo, denominado $G(V, C \cup D)$, em que V é o conjunto de vértices e representam as operações dos vários trabalhos, C é o conjunto de arcos conjuntivos que ligam as operações i e $i + 1$ de um trabalho, D é o conjunto de arcos disjuntivos que ligam as operações que requerem o mesmo recurso. Desta forma é escolhida ordem de processamento em cada recurso de forma a cumprir com as restrições e minimizando/maximizando o critério em estudo.

O modelo indexado no tempo, por sua vez, procura definir o instante de início de cada operação, sendo esta a variável de decisão que o modelo utilizará.

O modelo baseado em ordenação define explicitamente a sequência de operações que deverão ocorrer em cada operação.

Além do modelo, também será necessário um programa capaz de o resolver, como o Gurobi e o CPLEX, que utilizam algoritmos para encontrar a solução ótima. Em contra mão, é possível desenvolver algoritmos específicos a cada problema.

Existem vários algoritmos utilizados para este fim, como *Branch and Bound* e *Branch and Cut*, entre outros. Estes algoritmos são utilizados para reduzir o número de soluções a explorar ao não considerar soluções que certamente serão inferiores.

2.2.2 Heurísticas

Heurísticas são algoritmos de rápida resolução que devem permitir a obtenção de uma solução viável de forma rápida, por isso será de esperar que esta abordagem não implique a obtenção da solução ótima.
Frequentemente a formulação de uma heurística apenas funcionará para o problema para qual foi desenhada, devido à intuição apresentada na sua construção [11].

2.2.3 Meta-Heurísticas

Meta-heurísticas são o meio termo entre a qualidade das soluções apresentadas por algoritmos exatos e a rapidez das heurísticas. Existe uma grande quantidade de abordagens existentes, em que cada pertence a grupos diferentes. Podem ser baseadas em população, baseadas em trajetórias, com memória ou sem, e baseadas em processos naturais.

Algumas das meta-heurísticas com mais impacto são: *Simulated Annealing*, *Differential Evolution*, *Particle Swarm Optimization*, *Genetic Algorithm*, *Ant Colony Optimization*, e *Tabu Search Algorithm* [11]. Todas estas meta-heurísticas poderão ser utilizadas para resolver o

problemas apresentado e serão discutidas na próxima secção.

Simulated Annealing é uma meta-heurística evolucionária baseada no processo de arrefecimento de metais, em que se verifica uma menor energia do sistema quando este arrefecimento acontece lentamente. Desta forma, é gerada uma solução i com uma vizinhança S_i de onde é gerada uma nova solução j, que será avaliada consoante a função objetivo $f(j)$, a solução j poderá ser aceite mesmo não apresentando melhor valor de acordo com o algoritmo Metropolis [12]:

$$Pr\{\text{aceitar } j\} = \begin{cases} 1, & \text{se } f(j) < f(i) \\ e^{\frac{f(i)-f(j)}{c_k}}, & \text{caso contrário} \end{cases}$$

Onde c_k é a temperatura durante a iteração k. No início do algoritmo, quando a temperatura é alta, existe probabilidade acrescida de aceitar soluções piores, ao reduzirmos a temperatura com cada iteração, a probabilidade de aceitar soluções piores que a atual diminui.
Também será necessário definir como se realiza o arrefecimento, a diminuição de c_k, quantas vizinhanças procurar por iteração, e quando se dá por terminado o algoritmo. Cada vizinho visitado pode, ou não, representar uma solução viável, deve-se ter cuidado na escolha da representação da solução para que garantir a viabilidade da solução, ou relaxar as restrições ao incluir a punição sobre a sua violação na função objetivo.

Particle Swarm Optimization é uma meta-heurística populacional baseada no comportamento natural de animais de rebanho durante a procura de comida. Para tal são gerados P partículas, tal que a partícula i se encontra na posição $X^i(t)$ na iteração t, bem como uma velocidade $V^i(t)$. Com cada iteração a posição e velocidade de cada partícula é atualizada:

$$X^i(t+1) = X^i(t) + V^i(t+1)$$

$$V^i(t+1) = \omega V^i(t) + c_1 r_1 (pbest^i - X^i(t)) + c_2 r_2 (gbest - X^i(t))$$

Onde r_1 e r_2 são valores aleatórios entre 0 e 1, ω, c_1 e c_2 são os parâmetros da meta-heurística e chamam-se de inércia, coeficiente cognitivo e coeficiente social, respetivamente. $pbest^i$ é a melhor solução encontrada pela partícula i, $gbest$ é a melhor solução já encontrada.
Tal como *SA*, não existe a garantia inerente da viabilidade de cada solução, deve-se por isso utilizar um algoritmo que repare cada solução ou relaxar restrições para a função objetivo, com um peso sobre tal. Também será necessário decidir quando terminar a procura de novas soluções, poder-se-à utilizar um conceito semelhante ao descrito anteriormente para *SA*.

3

Revisão de Literatura

Este capítulo irá explorar a literatura existente em relação ao mapeamento de processos e sobre as abordagens utilizadas para a resolução do problema apresentado.

3.1 Abordagens para a Resolução

Nesta secção será apresentada uma amostra da literatura existente sobre as abordagens existentes para a resolução dos problemas *Job-Shop* e os seus derivados. A literatura aqui apresentada não será exaustivas, mas demonstrará algumas tendências existentes para problemas próximos ao apresentado nesta dissertação.

O critério mais estudo nesta área de investigação é a minimização do *makespan* [13], mesmo assim existem alguns critérios diferentes que têm aparecido na literatura. Desta forma, pretende-se dar a conhecer alguns destas outras vertentes.

3.1.1 Algoritmos Exatos

Algoritmos exatos foram a principal abordagem utilizada para resolver problemas de *Job-Shop* mas têm vindo a tornar-se menos considerados com o aumento do número de extensões e restrições que os novos problemas apresentam [10].

Como já referido, existem três grandes modelos utilizados neste espaço. O modelo disjuntivo, o modelo indexado no tempo, e o modelo baseado em ordenação. Em [14] são avaliados os modelos de acordo com a sua eficiência computacional, onde se observa que o modelo disjuntivo é o mais eficiente a resolver problemas *Job-Shop* de pequenas e grandes dimensões. Será razoável esperar que este facto seja verdade para outras vertentes do problema.

Em [4] é apresentado uma formulação do modelo disjuntivo utilizando *MILP* para resolver o problema *Flexible Job-Shop* com o objetivo de minimizar o *makespan*.
Existem também artigos que geram ou criam problemas que se tornam em *data-sets*, sendo utilizados em vários artigos de forma a comparar resultados. Um exemplo é o *data-set SFJS* e *MFJS* criado por Fattahi et al. [15], onde também se apresenta uma formulação para resolver o problema *Flexible Job-Shop*. Em Ozbakir et al. [16] é apresentada outra formulação baseada no modelo disjuntivo, ao mesmo tempo é comparada com o modelo de Fattahi et al. [15] em relação à eficiência computacional, verificando que esta formulação é superior na maioria do *data-set*. Em Thörnblad et al. [17] é apresentada e comparada uma formulação baseada no modelo indexado no tempo, onde se observa a inferior eficiência deste modelo.

Samarghandi [18] propõe um modelo *MILP* para a resolução do problema *No-wait Job-Shop* com o objetivo de minimizar o *makespan* com a restrição adicional de não ultrapassar a data limite de cada trabalho. Bem como dois modelos de *Constraint Programming* que apresentam boas soluções em relação ao modelo *MILP* em problemas de grande dimensão.

Em Behmanesh et al. [19] é apresentado um modelo *MILP* para resolver o problema *Flexible Multi-Resource Job-Shop*.

3.1.2 Heurísticas

A heurística Nawaz-Enscore-Ham [20] é tida como a heurística a superar, desenvolvida par o problema *Flow-Shop*, mas que já foi utilizada em vários artigos, especialmente na criação de soluções iniciais [21], [22].

3.1.3 Meta-Heurísticas

A escolha correta das variáveis de *SA* é de grande importância para que se obtenha boas soluções com pouco desperdício computacional, existem cinco variáveis: A temperatura inicial c_0, o critério de paragem CP, o número de vizinhos visitados por iteração L_k, a função de redução da temperatura α, e quando aplicável a punição a aplicar sobre a função objetivo por violação de restrições.
Existem várias formas de estimar a temperatura inicial, idealmente será alta o suficiente de forma a evitar a influência da solução inicial ao aceitar a maioria dos vizinhos propostos. Delahaye et al. [23] descreve uma forma de estimar este parâmetro. Definir uma temperatura baixa e um rácio a superar $\chi(c_0)$, realizar L_k visitas na vizinhança e calcular o rácio entre movimentos aceites e não aceites, se esse rácio for maior que o definido então a temperatura inicial foi alcançada, caso contrário a nova temperatura será βc_0, com $\beta > 1$.
Outra vez, Delahaye et al. [23] descreve um possível critério de paragem. Deve-se dar como terminada a procura quando não ocorre a melhoria da melhor solução durante várias iterações consecutivas. Quanto mais iteração mais provável é encontra uma solução mais

próxima do ótimo, o que por sua vez também aumenta o tempo computacional utilizado. Semelhantemente, o número de visitas L_k deve ser alto o suficiente para que se obtenha melhores soluções, sem que seja tão elevado de forma a aumentar desproporcionalidade o tempo computacional.
A função de redução da temperatura pode tomar várias formas, em que a mais simples será aqui utilizada, a temperatura na iteração k é dada por $c_k = \alpha c_{k-1}$, em que tipicamente $0.8 < \alpha < 0.99$. Existe a garantia que a procura tende para o mínimo global quando a redução da temperatura é infinitamente longa e infinitamente pequena entre iteração, mas como é de esperar isto não é praticável na realidade.

Xia et al. [24] utiliza um algoritmo híbrido entre *PSO* e *SA* na resolução do problema *Flexible Job-Shop* com tempos de processamento diferentes dentro de cada recurso-tipo. Onde *PSO* é utilizada na atribuição de operações às máquinas, enquanto *SA* é utilizada para escalonar operação a cada recurso. Jonathan et al. [25] comparam quatro meta-heurísticas para resolve o problema *Flexible Flow-Shop with No-Wait*. A meta-heurística proposta, *Whale Optimization Algorithm* gera solução com qualidade semelhante a *SA*, *Genetic Algorithm* e *Minimum Deviation Algorithm*, contudo *MDA* demonstra tempo computacional muito menor.

Ying et al. [21] comparam quatro variantes de *SA* na resolução do problema *Job-Shop with No-Wait*. Demonstram que a utilizam adicional do retorno há melhor solução e a existência de várias soluções iniciais, permite a obtenção de melhores soluções no mesmo espaço de tempo.

Perrachon et al. [26] sugerem a utilização de *SA* para a otimização do problema *Flexible Multi-Resource Job-Shop*. Para tal são exploradas oito diferentes estruturas da vizinhança baseadas no modelo disjuntivo. A restrição *Blocking* encontra-se relaxada neste problema, ao ser possível que vários recursos sejam necessários durante fases diferentes de cada operação, podendo ser visto como a decomposição da operação em várias com *No-Wait*.

Caumond et al. [27] apresenta uma meta-heurística baseada no modelo disjuntivo, utilizando *Memetic Algorithm* para a resolução do problema *Job-Shop with No-Wait* e *Job-Shop with Maximum Time Lag*.

4

Mapeamento dos Exames

O processo de mapeamento dos exames começou com a decisão de quais incluir no estudo. Nem todos os exames foram considerados devido à baixa frequência com que estes ocorrem e a consequente falta de dados em relação à duração das suas atividades. Desta forma, apenas foram mapeados aqueles que ocorrem pelos menos mensalmente. De seguida, foi necessário definir os recursos-tipo existentes no sistema, estes podem ser divididos em recursos humanos, técnicos auxiliares de saúde (TAS), enfermeiros, técnicos superiores de diagnóstico e terapêutica de medicina nuclear (TSDT), médicos especialistas (ME), e médicos cardiologistas (MC); e divididos em recursos físicos, gabinetes médicos (GB), sala de espera (SE), sala de espera de crianças (SEC), sala cortinas (SC), sala polivalente (SP), sala enfermagem 1 (SE1), sala enfermagem 2 (SE2), sala enfermagem 3 (SE3), sala câmara gama (SCG), e sala tomografo (ST).

Tabela 4.1: Capacidade dos recursos humanos

Recurso-tipo	TAS	Enfermeiros	TSDT	ME	MC
Capacidade	4	4	4	2	1

Tabela 4.2: Capacidade dos recursos físicos

Recurso-tipo	GM	SE	SEC	SC	SP	SE1	SE2	SE3	SCG	ST
Capacidade	2	4	3	2	1	1	3	3	1	1

Cada recurso-tipo tem uma quantidade de recursos disponíveis, para os recursos humanos corresponde ao número de profissionais em cada grupo existentes no departamento, para os recursos físico corresponde à lotação de pacientes que podem suportar, iremos considerar que existe portanto uma capacidade máxima para cada recurso-tipo, estas capacidades estão apresentadas na Tabela 4.1 e Tabela 4.2, respetivamente. Admite-se que cada recurso atribuído a uma tarefa será utilizado durante a duração total da atividade.

Tabela 4.3: Capacidade dos recursos fictícios

Recursos fictícios	SP	SE1	SP ou SE1	SE2	SE2 ou SE3
Capacidade	1	1	2	1	6

Existem recursos-tipo que podem ser equiparados a outros em certas atividades. A sala polivalente e sala enfermagem 1 podem ser tidas como equivalentes em certas atividades. Por isso foi necessário criar recursos-tipo fictícios. Para o exemplo anterior, foi criado o recurso-tipo sala polivalente ou sala enfermagem 1, sempre que uma destas salas é explicitamente necessária, a sua utilização é representadas no recurso-tipo correspondente e no recurso-tipo fictício, caso seja indiferente a sala necessárias, a utilização apenas é representada no recurso-tipo fictício. Contrariamente, a sala enfermagem 2 e sala enfermagem 3 podem também ser tidas como equiparadas, mas quando a sala enfermagem 2 é explicitamente necessária a sua capacidade é menor, passando de 3 para 1 pessoa, por isso o recurso fictício terá uma utilização de 3 lugares quando a sala de enfermagem 2 é necessária. Ambos os casos aparecem na Tabela 4.3, onde se observa a capacidade que será utilizada na sala de enfermagem 2.

Num passo seguinte, cada exame foi decomposto nas suas atividades. Considerou-se que uma nova atividade deve começar quando existe uma mudança dos recursos necessários. Foram então realizadas entrevistas de forma a realizar a decomposição dos exames, no seguimento das entrevistas as atividades foram validadas pelos vários grupos profissionais. Rapidamente identificou-se a necessidade de diferenciar o mapeamento entre pacientes com mobilidade autónoma, e aqueles com mobilidade condicionada, e crianças, existindo diferenças nos recursos necessários e na duração de cada atividade.
Seguidamente, as duração de cada atividade foram recolhidas utilizando folhas de registo provenientes do mapeamento realizado. A duração média de cada atividade foi obtida através da hora de início e de fim de cada atividade registadas pelos profissionais, será de salientar que mesmo sendo este um problema de *No-Wait*, nem sempre foi verificado nas folhas de registo, evidenciando um possível relaxamento da restrição.
Todas as folhas de registo e respetivos dados estarão em anexo com esta dissertação.

5

Desenvolvimento dos Modelos

Este capítulo os três problemas que devem ser resolvidos, para tal foram desenvolvidos vários modelos. No primeiro problema existem recursos e exames fixos, sendo o objetivo minimizar o tempo entre o começo e o fim de todos os exames, chamado de *makespan*. O segundo problema tem recursos e tempo fixos, sendo o objetivo maximizar o número de exames a realizar, utilizando um somatório simples ou ponderado. Finalmente, o terceiro problema é caracterizado por exame e tempos fixos, sendo o objetivo minimizar os recursos necessários para realizar todos os exames no tempo definido.

5.1 Problema de *Makespan*

Este problema irá utilizar o exemplo de uma segunda-feira típica em termos de exames realizados. Ocorrendo três cintigrafia tiroideia, cinco cintigrafia pulmonar de ventilação/inalação + perfusão, dez cintigrafia miocárdica de perfusão em repouso, uma cintigrafia das glândulas salivares, e dez PET-FDG.
A próxima formulação é a que tradicionalmente se encontra quando indexada no tempo, chamá-la-emos de *MILP-trad*:

Conjuntos:
O conjunto de trabalhos $I, i \in I := (1, \ldots, n)$
O conjunto de operações $K, k \in K := (1, \ldots, K_{\max})$
O conjunto de recursos-tipo $R, p \in R := (1, \ldots, R_{\max})$
O conjunto de instantes de tempo $T, t \in T := (1, \ldots, T_{\max})$

Parâmetros:
$\rho_{i,k}$ é o tempo de processamento da operação k do trabalho i
$r_{i,p,k}$ é o número de recursos p necessários para realizar a operação k do trabalho i
C_p é a capacidade do recursos do tipo p

Variáveis de Decisão:

$X_{t,i,k}$ é uma variável binária com valor 1 se a operação k do trabalho i ocorrer durante o instante t, caso contrário tem valor 0
$Z_{t,i,k}$ é uma variável binária com valor 1 se a operação k do trabalho i começar no instante t, caso contrário tem valor 0
$S_{i,k}$ é uma variável inteira que representa o valor de começo da operação k do trabalho i
$F_{i,k}$ é uma variável inteira que representa o valor de fim da operação k do trabalho i
$C_{\max}$ é uma variável inteira que representa o *makespan*

Função Objetivo:

$$\min C_{\max} \tag{5.1}$$

Sujeito a:

$$F_{i,K_{\max}} \leq C_{\max} \quad \forall i \tag{5.2}$$

$$\sum_{t} Z_{t,i,k} = 1 \quad \forall i,k \tag{5.3}$$

$$\sum_{t} X_{t,i,k} = \rho_{i,k} \quad \forall i,k \tag{5.4}$$

$$\sum_{t'=t}^{t+\rho_{i,k}} X_{t',i,k} \geq \rho_{i,k} Z_{t,i,k} \quad \forall i,k,t = 1,\ldots,T-\rho_{i,k} \tag{5.5}$$

$$S_{i,k} = \sum_{t} t Z_{t,i,k} \quad \forall i,k \tag{5.6}$$

$$S_{i,k+1} = F_{i,k} \quad \forall i,k \tag{5.7}$$

$$F_{i,k} - S_{i,k} = \rho_{i,k} \quad \forall i,k \tag{5.8}$$

$$\sum_{i}\sum_{k} r_{i,p,k} X_{t,i,k} \leq C_p \quad \forall t,p \tag{5.9}$$

A função objetivo (5.1) minimiza o *makespan* dos trabalho, ou seja, o tempo entre o início do primeiro exame, e o fim do último.
A restrição (5.2) garante que todos os trabalho acabam antes ou no instante do *makespan*, auxiliando a função objetivo.
A restrição (5.3) garante que só existe um instante em que a operação k do trabalho i é atribuído.
A restrição (5.4) garante que para a operação k do trabalho i o somatório de $X_{t,i,k}$ sobre t tem o valor da duração $\rho_{i,k}$.
A restrição (5.5) garante a concordância do posicionamento sobre t entre $X_{t,i,k}$ e $Z_{t,i,k}$.
A restrição (5.6) garante a coerência do momento de início da tarefa k.
A restrição (5.7) garante que para o trabalho i a operação $k+1$ começa quando a operação k acaba, verificando a restrição de *No-Wait*.
A restrição (5.8) garante que a duração da operação k para o trabalho i é de $\rho_{i,k}$.
A restrição (5.9) garante que não existe a sobre-utilização do recursos p durante o instante

*t*m verificando as extensões *Multi-Resource* e *Flexible*.

Por outro lado, é possível formular modelos mais eficientes que este clássico. Isto poderá ser feito com o pré-processamento de cada trabalho, permitindo inserir cada trabalho na sua totalidade e verificar se este irá causar sobre-utilização de recursos, chamá-la-emos de *MILP-δ*.

Por outro lado, será possível formular modelos mais eficientes que o clássico. A formulação aqui sugerida, *MILP-δ*, tira partido do facto da restrição *No-Wait* criar um padrão de utilização de recursos, qualquer que seja o instante t onde este se insere. Desta forma não é necessário explicitamente formular restrições que garantem *No-Wait*.

Conjuntos:
O conjunto de trabalhos $I, i \in I := (1, \ldots, n)$
O conjunto de recursos-tipo $R, p \in R := (1, \ldots, R_{\max})$
O conjunto de instantes de tempo $T, t \in T := (1, \ldots, T_{\max})$

Parâmetros:
ρ_i é duração do trabalho i
$\delta_i(u, p)$ é a quantidade de recursos do tipo p necessários a um offset de u instantes de tempo no trabalho i
C_p é a capacidade do recursos do tipo p

Variáveis de Decisão:
$Z_{t,i}$ é uma variável binária com valor 1 se o trabalho i começar no instante t, caso contrário tem valor 0
F_i é uma variável inteira que representa o valor de fim do trabalho i
$C_{\max}$ é uma variável inteira que representa o *makespan*

Função Objetivo:

$$\min C_{\max} \tag{5.10}$$

Sujeito a:

$$F_i \leq T_{\max} \quad \forall i \tag{5.11}$$

$$\sum_{t=0}^{T_{\max}-\rho_i+1} Z_{t,i} = 1 \quad \forall i \tag{5.12}$$

$$\sum_{t=0}^{T_{\max}-\rho_i+1} Z_{t,i} * (t + \rho_i) = F_i \quad \forall i \tag{5.13}$$

$$\sum_{i} \sum_{\tau=\max(0,t-\rho_i+1)}^{\min(t,T_{\max}-\rho_i)} \delta_i(t-\tau,p) Z_{\tau,i} \leq C_p \quad \forall t,p \tag{5.14}$$

A função objetivo (5.10) minimiza o *makespan* dos trabalho.
A restrição (5.11) garante que todos os exames acabam antes ou no instante do *makespan*, auxiliando a função objetivo.
A restrição (5.12) garante que só existe um instante de começo do trabalho i.
A restrição (5.13) faz a ligação entre as variáveis $Z_{t,i}$ e F_i.
A restrição (5.14) garante que não há sobre utilização de recursos, ao verificar para cada instante t quais trabalhos estão a ser executados e em que fase este se encontra, garantindo que em cada instante não há sobre-utilização de recursos.

Para ambas das formulações apresentadas, o tamanho do conjunto de instantes de tempo, T, tem impacto no número de variáveis e restrições, será por isso importante encontrar o menor valor de $T_{\max}$ possível mas que seja maior que *makespan*, a isto tipicamente chama-se de *upper-bound*. No decorrer desta secção demonstrar-se-à o impacto que este valor tem nas formulações apresentadas.

De seguida vão ser apresentados dois modelos diferentes utilizando *Simulated Annealing*, sendo esta uma meta-heurística podemos garantir a viabilidade de cada solução na procura da vizinhança ou permitir o relaxamento de restrições ao coloca-las na função objetivo. No problema aqui discutido, a única restrição que poderá ser relaxada é a sobre-utilização de recursos.

O primeiro modelo apresenta sem dúvida a codificação mais simples, utilizando o momento inicial de cada exame como a solução do problema, facilitando a obtenção de qualquer informação a partir deste valor. Sendo assim, a vizinhança de uma solução é obtida pela deslocação temporal do início de um trabalho. Contudo, poderá por em causa a sobre-utilização de recursos.

```
[808, 255, 287, 9, 274, 16, 49, 400, 260, 186, 215, 287, 207, 762, 207, 83, 204, 542, 28, 879, 619, 669, 572]
```

Figura 5.1: Codificação do modelo 1

Uma solução vizinha poderá ser obtida de forma aleatória em que apenas é necessário definir o trabalho a deslocar e o momento no qual este deve começar, não havendo a garantia de viabilidade, este modelo será $M1_{NGV}$. Outra hipótese é verificar todos os momentos no qual se pode inserir um trabalho sem sobre-utilizar recursos e escolher, aleatoriamente, um desses momentos, este modelo será $M1_{GV}$.

O segundo modelo é codificado pela sequência pela qual se agenda cada trabalho, serão necessários passos adicionais para obter a informação obtida no primeiro modelo. Para obter a vizinhança basta modificar a sequência dos exames a agendar, sendo assim é necessário um algoritmo extra que descodifique a solução.

```
[1, 18, 0, 11, 12, 2, 6, 14, 8, 4, 19, 13, 9, 3, 7, 20, 22, 15, 16, 5, 21, 10, 17]
```

Figura 5.2: Codificação do modelo 2

Este tipo de modelo é decomposto em dois subproblemas, o problema de sequenciamento e o problema de escalonamento. Primeiro define-se a sequencia de exames e depois procura-se qual deve o momento de começo de cada um. Este primeiro subproblema é tipicamente resolvido utilizando meta-heurísticas, ao contrário do segundo sub-problema que tipicamente é resolvido com algoritmos.
Existe um conjunto de algoritmos concebidos para este fim, *Non-delay timetabling* [28] , *Enhanced timetabling* [29], e *Shift timetabling* [30] . Em Zhu et al. [30] são comparados estes três algoritmos relativamente há qualidade das soluções geradas, chegando à conclusão que *Shift timetabling* é a que apresenta as melhores soluções.
Este algoritmo tenta escalonar sequencialmente cada trabalho sem verificar $t_{\pi_i} \geq t_{\pi_{i-1}}$, chamada de restrição ordinária. Ou seja, na sequência $\pi = (1, 2, 3)$ nada impede que o trabalho 3 comece antes do trabalho 2.
Qualquer solução obtida é garantidamente viável devido ao algoritmo de agendamento que descodifica a solução.

Para estes diferentes modelos os parâmetros utilizados têm grande impacto na qualidade das soluções alcançadas e no tempo computacional necessário para as alcançar. Desta forma é preciso estudar o nível de cada variável que mais se adequa para cada modelo, para tal na Tabela 5.1 os vários níveis são apresentados.

Tabela 5.1: Níveis das variáveis dos modelos de *Simulated Annealing*

Modelo	$\chi(c_0)$	CP	L_k	α	P
$M1_{GNV}$	0.5, 0.7, 0.9	10, 55, 75	500, 1500, 2500	0.8, 0.9, 0.975	10, 55, 100
$M1_{GV}$	0.5, 0.7, 0.9	5, 30, 55	10, 30, 50	0.8, 0.9, 0.975	-
$M2$	0.5, 0.7, 0.9	5, 30, 55	10, 30, 50	0.8, 0.9, 0.975	-

Os diversos níveis apresentados foram escolhidos como valores razoáveis para tal. L_k

e CP diferem entre os modelos de forma a manter tempo computacional semelhante entre estes. Os níveis para peso de violação de restrições foram escolhidos com algumas experiências preliminares. O impacto de cada combinação na função objetivo é demonstrada na Figura 5.3.

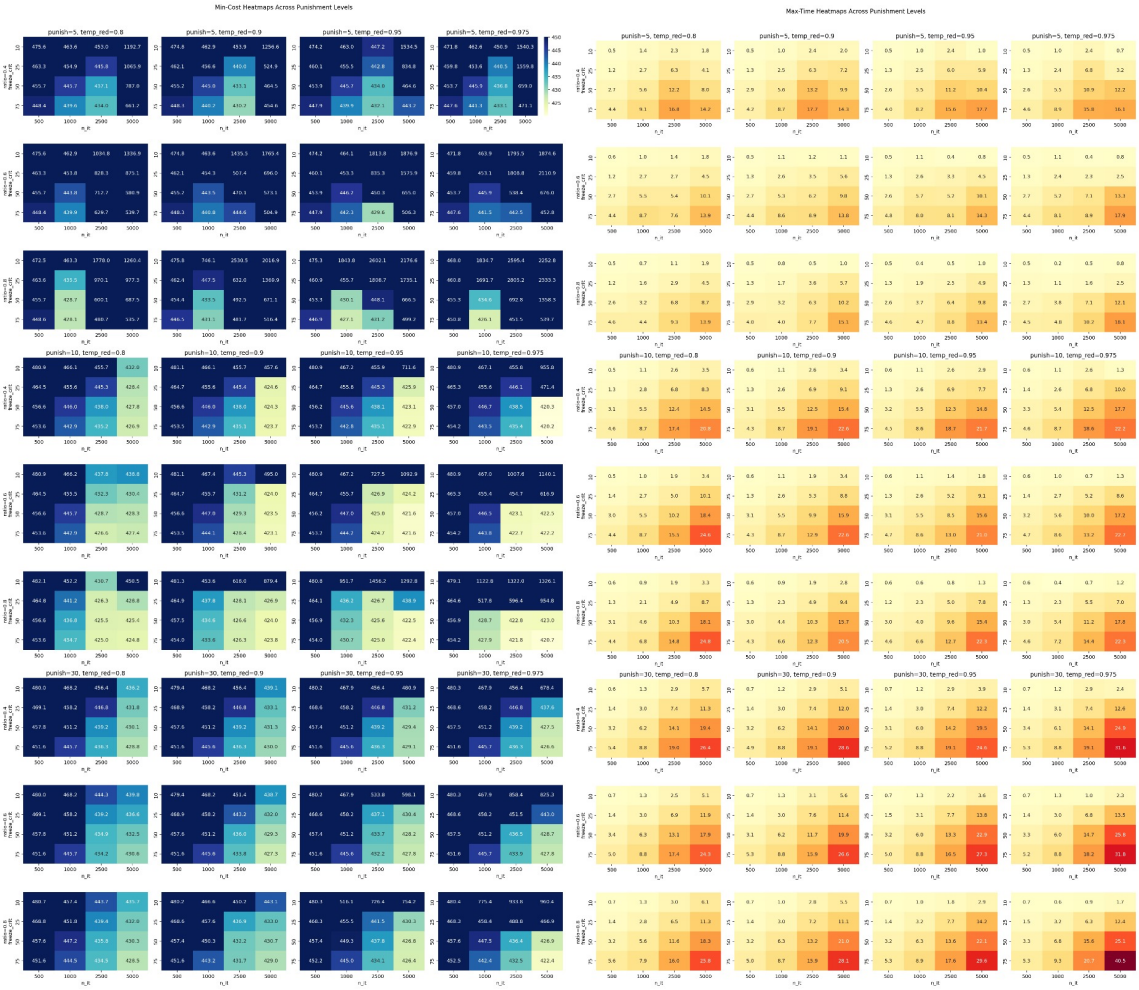

Figura 5.3: Impacto na qualidade e tempo computacional das combinação dos níveis para o Modelo 1 NGV

Para cada valor apresentado foram realizadas 100 **corridas**, utilizando o paralelismo oferecido por processadores modernos, foi possível realizar efetivamente 10 **corridas** com 10 amostras cada. Desta forma o valor reportado é a média do mínimo das 10 amostras. Semelhantemente o tempo computacional reportado é a média do máximo das 10 amostras.

Observamos de imediato o grande impacto que a escolha correta dos níveis de cada variável tem na qualidade da solução e no tempo computacional necessária, apresentando valores com grande variação. Soluções melhores necessitam de mais tempo para serem alcançadas, mas o oposto nem sempre se verifica.

A escolha da melhor combinação de variáveis é dificultado por existirem duas variáveis de resposta que têm de ser consideradas, de qualquer forma a combinação $\{0.8, 25, 2500, 0.8, 10\}$, $\{\chi(c_0), CP, L_k, \alpha, P\}$.

Para o modelo 1 GV os valores obtidos são reportados na Figura 5.4.

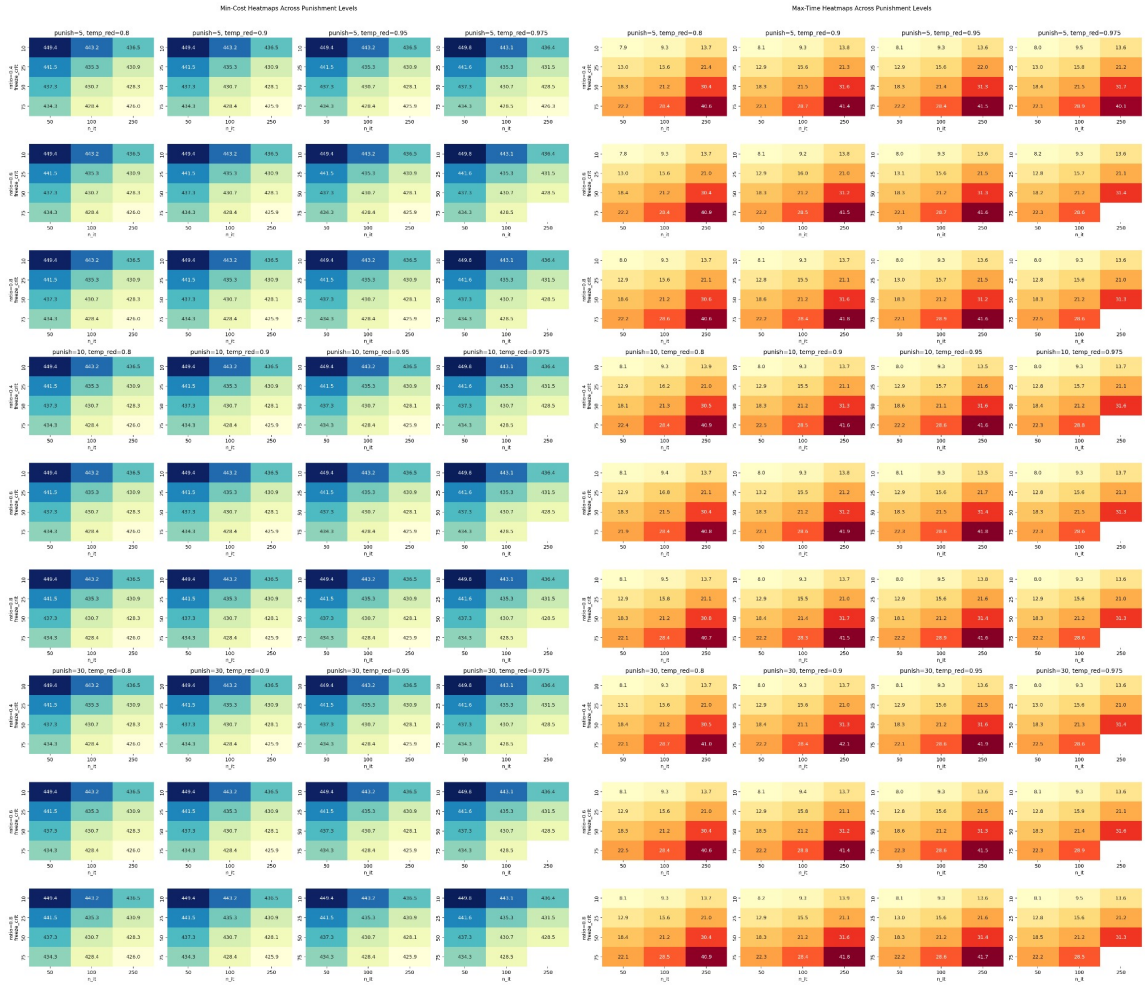

Figura 5.4: Impacto na qualidade e tempo computacional das combinação dos níveis para o Modelo 1 GV

Ao contrário do modelo 1 NGV, o impacto de $\chi(c_0)$ e de α é mínimo em relação à qualidade da solução e ao tempo computacional. Como será de esperar, L_k e CP continuam a ser os grandes decisores sobre a qualidade da solução e tempo computacional. Se pretendermos obter tempo computacional semelhante à do modelo anterior, a combinação a escolher será $\{\mathbf{0.4}, 10, 50, \mathbf{0.8}, -\}$.

Finalmente, o modelo 2 tem os seus valores reportados na Figura 5.5.

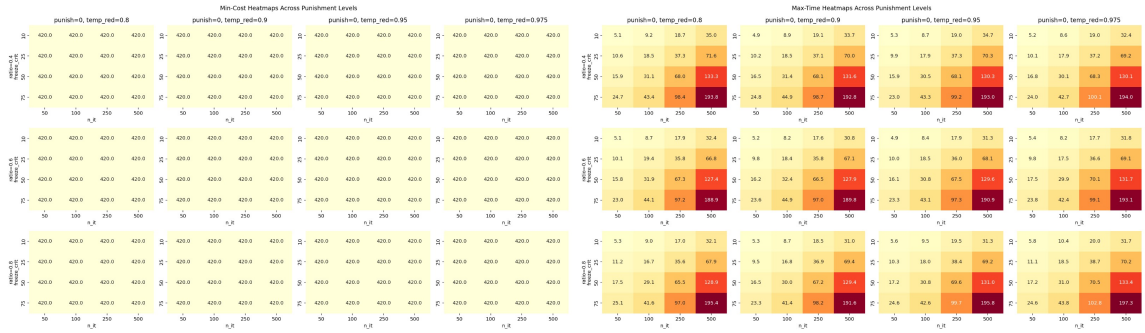

Figura 5.5: Impacto na qualidade e tempo computacional das combinação dos níveis para o Modelo 1 GV

É imediatamente óbvio que nenhuma das variáveis tem um impacto na qualidade das soluções, contudo tem no tempo computacional necessário. Novamente $\chi(c_0)$ e α demonstram pouca importância para a escolha a fazer, por sua vez ao aumentar L_k e CP também aumenta o tempo computacional. Uma boa combinação a escolher será $\{\mathbf{0.4}, 10, 50, \mathbf{0.8}, -\}$.

Finalmente será possível comparar os resultados obtidos pelas formulações e modelos apresentados. Estes resultados encontram-se na Tabela 5.2. Para a obtenção dos resultados foi utilizado um Ryzen 5600, Gurobi 12.0 para ambas as formulações *MILP*, e *Python* 3.11 para a implementação dos modelos de *SA*, quando nada é dito sobre o contrário.

Tabela 5.2: Comparação da qualidade da solução e do tempo computacional em relação ao modelo e formulação utilizados

Modelo	Combinação	$T_{\max}$	Valor Objetivo	*Best Bound*	Tempo Computacional (s)
MILP-trad	-	3685	-		28800
MILP-trad	-	420	419.0	419	28800
MILP-δ	-	3685	415.0	415	61.7
MILP-δ	-	420	415.0	415	11.3
M1 NGV	$\{0.8, 25, 2500, 0.80, 10\}$	-	426.3	-	4.6
M1 NGV (C)	$\{0.4, 75, 5000, 0.95, 10\}$	-	420.1	-	2.9
M1 GV	$\{\mathbf{0.4}, 10, 50, \mathbf{0.80}, -\}$	-	449.9	-	7.9
M2	$\{\mathbf{0.4}, 10, 50, \mathbf{0.80}, -\}$	-	420.0	-	5.1

Todos os modelos apresentados são de razoável qualidade, contudo é possível realçar algumas diferenças entre estes. *MILP-δ* apresenta uma solução ótima numa quantidade de tempo razoável, este facto é especialmente verdade quando o valor de $T_{\max}$ utilizado é já próximo do valor ótimo do *makespan*. Por outro lado, é difícil justificar a utilização de *MILP-trad* em qualquer contexto real, ao necessitar de várias horas para resolver o problema apresentado.

Todos os modelos baseados em *SA* apresentam boas soluções requerendo pouco tempo computacional. O modelo 2 demonstra a melhor solução alcançada dentro deste conjunto,

contudo é aparente a limitação existente na codificação do problema ou no algoritmo de descodificação utilizado, visto que qualquer a combinação de variáveis utilizadas, a solução nunca será melhor que 420. Por sua vez, o modelo 1 NGV permite alcançar soluções melhores que 420, mesmo que para tal seja necessário tempo computacional cada vez maior, para tal a utilização de outras linguagens de programação permitirá reduzir o tempo computacional utilizado. Finalmente o modelo 1 GV fica à quem dos anteriores, não demonstrando soluções tão boas e elevado tempo computacional.

5.2 Problema do Número de Exames

Como na secção anterior, foi definido um exemplo para se comparar os modelos propostos. Este é definido por $[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 5]$ que correspondem aos mesmo exames anteriormente exemplificado, de modo a serem agendados dentro de um período de 480 minutos.

Para este problema, o critério a otimizar já é algo menos comum, dado que o *makespan* já é definido, pretende-se por isso maximizar o número de exames que se realizam no período em estudo, nomeadamente durante o horário laboral do departamento de Medicina Nuclear. Facilmente se modifica a formulação tradicional, *MILP-trad*, de modo a otimizar o pretendido.

Conjuntos:
O conjunto de trabalhos $I, i \in I := (1, \ldots, n)$
O conjunto de operações $K, k \in K := (1, \ldots, K_{\max})$
O conjunto de recursos-tipo $R, p \in R := (1, \ldots, R_{\max})$
O conjunto de instantes de tempo $T, t \in T := (1, \ldots, T_{\max})$

Parâmetros:
$\rho_{i,k}$ é o tempo de processamento da operação k do trabalho i
$r_{i,p,k}$ é o número de recursos p necessários para realizar a operação k do trabalho i
C_p é a capacidade do recursos do tipo p

Variáveis de Decisão:
Y_i é uma variável binária com valor 1 se o trabalho i estiver agendado, caso contrário tem valor 0
$X_{t,i,k}$ é uma variável binária com valor 1 se a operação k do trabalho i ocorrer durante o instante t, caso contrário tem valor 0
$Z_{t,i,k}$ é uma variável binária com valor 1 se a operação k do trabalho i começar no instante t, caso contrário tem valor 0
$S_{i,k}$ é uma variável inteira que representa o valor de começo da operação k do trabalho i

$F_{i,k}$ é uma variável inteira que representa o valor de fim da operação k do trabalho i

Função Objetivo:

$$\max \sum_i Y_i \tag{5.15}$$

Sujeito a:

$$F_{i,K_{\max}} \leq T_{\max} \quad \forall i \tag{5.16}$$

$$\sum_t Z_{t,i,k} = Y_i \quad \forall i,k \tag{5.17}$$

$$\sum_t X_{t,i,k} = \rho_{i,k} Y_i \quad \forall i,k \tag{5.18}$$

$$\sum_{t'=t}^{t+\rho_{i,k}} X_{t',i,k} \geq \rho_{i,k} Z_{t,i,k} \quad \forall i,k,t = 1,\ldots,T-\rho_{i,k} \tag{5.19}$$

$$S_{i,k} = \sum_t t Z_{t,i,k} \quad \forall i,k \tag{5.20}$$

$$S_{i,k+1} = F_{i,k} \quad \forall i,k \tag{5.21}$$

$$F_{i,k} - S_{i,k} = \rho_{i,k} Y_i \quad \forall i,k \tag{5.22}$$

$$\sum_i \sum_k r_{i,p,k} X_{t,i,k} \leq C_p \quad \forall t,p \tag{5.23}$$

A função objetivo (5.15) maximiza o número de trabalho que são escalonados.
A restrição (5.16) garante que todos os trabalho acabam antes do *makespan*.
A restrição (5.17) garante que só existe um instante de começo da tarefa k do trabalho i e se este estiver atribuído.
A restrição (5.18) garante que para a operação k do trabalho i o somatório de $X_{t,i,k}$ sobre t tem o valor da duração $\rho_{i,k}$ se o trabalho i estiver atribuído.
A restrição (5.19) garante a concordância do posicionamento sobre t entre $X_{t,i,k}$ e $Z_{t,i,k}$.
A restrição (5.20) garante a coerência do momento de início da tarefa k.
A restrição (5.21) garante que para o trabalho i a operação $k+1$ começa quando a operação k acaba, verificando a restrição de *No-Wait*.
A restrição (5.22) garante que a duração da operação k para o trabalho i é de $\rho_{i,k}$ só se o trabalho i estiver atribuído.
A restrição (5.23) garante que não existe a sobre-utilização do recursos p durante o instante tm verificando as extensões *Multi-Resource* e *Flexible*.

Outra vez, é possível modificar a formulação já apresentada de forma a otimizar este novo critério, segue-se assim *MILP-*δ.

Conjuntos:
O conjunto de trabalhos $I, i \in I := (1,\ldots,n)$
O conjunto de recursos-tipo $R, p \in R := (1,\ldots,R_{\max})$

O conjunto de instantes de tempo $T, t \in T := (1, \ldots, T_{\max})$

Parâmetros:
ρ_i é duração do trabalho i
$\delta_i(u, p)$ é a quantidade de recursos do tipo p necessários a um offset de u instantes de tempo no trabalho i
C_p é a capacidade do recursos do tipo p

Variáveis de Decisão:
$Z_{t,i}$ é uma variável binária com valor 1 se o trabalho i começar no instante t, caso contrário tem valor 0
Y_i é uma variável binária com valor 1 se o trabalho i estiver agendado, caso contrário tem valor 0

Função Objetivo:

$$\max \sum_i Y_i \tag{5.24}$$

Sujeito a:

$$\sum_{t=0}^{T_{\max}-\rho_i+1} Z_{t,i} = Y_i \quad \forall i \tag{5.25}$$

$$\sum_i \sum_{\tau=\max(0,t-\rho_i+1)}^{\min(t,T_{\max}-\rho_i)} \delta_i(t-\tau, p) Z_{\tau,i} \leq C_p \quad \forall t, p \tag{5.26}$$

A função objetivo (5.24) maximiza o número de trabalhos que são escalonados.
A restrição (5.25) garante que só existe um instante de começo do trabalho i e se este estiver atribuído.
A restrição (5.26) garante que não há sobre utilização de recursos, ao verificar para cada instante t quais trabalhos estão a ser executados e em que fase este se encontra, garantindo que em cada instante não há sobre-utilização de recursos.

Ao contrário do problema anterior, definir $T_{\max}$ é redundante visto que este valor será o do período no qual os exames são agendados.

De seguida são ser modificados os modelos utilizados no problema anterior de modo a conseguirem otimizar o novo critério e tendo em consideração as diferentes restrições apresentadas.

A primeira codificação permite identificar o momento de começo de cada exame, se este momento for identificado com -1 significará que o respetivo exame não será agendado

e não irá contribuir para a função objetivo.

```
[198, 261, -1, -1, -1, 10, 140, 96, 292, 66, 39, -1, 111, -1, 164, 20, 80, 51, 253, 122, 228, 307, 204, 86, -1, 151, 178,  4, 331, 282, 58, 30]
```

Figura 5.6: Codificação do modelo 1

A forma como se obtém a vizinhança também será diferente, terá de ser possível deslocar temporalmente trabalhos, mas também removê-los e adicioná-los, esta escolha é feita de forma aleatória com probabilidade definida previamente, adicionando assim duas novas variáveis, nomeadamente nei_1 e nei_2.

Como no problema anterior, não há uma garantia inerente da viabilidade da solução, guiamos a procura no sentido da viabilidade ao punir a sobre-utilização de recursos. Desta vez, como existem três tipos de vizinhança, será interessante saber se devemos garantir a viabilidade na procura de certas vizinhanças. Ou seja, o modelo NNVG não irá garantir a viabilidade de qualquer solução, NGV irá apenas garantir a viabilidade durante a adição de novos trabalhos, GV irá garantir a viabilidade de todas as soluções propostas.

Uma outra codificação possível é através da sequência de exames, em que apenas são apresentados os exames a agendar.

```
[0, 5, 11, 2, 20, 3, 12, 24, 27, 8, 17, 10, 6, 31, 19, 18, 21, 16, 1, 25, 29, 15, 30, 7, 23, 26, 4]
```

Figura 5.7: Codificação do modelo 2

Com esta codificação também será necessário 3 tipos diferentes de vizinhança, trocar a ordem dos trabalhos, removê-los e adicioná-los.

Contudo, não fica garantida a viabilidade da solução. Para tal é necessário guiar a procura através da punição da sobre-utilização de tempo, de forma a que os trabalhos agendados se encontrem todos no período definido.

O último modelo utiliza outra vez a ordem dos trabalhos.

```
[31, 11, 22, 0, 9, 1, 28, 7, 13, 14, 19, 24, 16, 12, 2, 25, 27, 10, 6, 18, 5, 17, 8, 21, 26, 15, 20, 23, 30, 29, 3, 4]
```

Figura 5.8: Codificação do modelo 3

Esta codificação permite utilizar apenas um tipo de vizinhança, sendo esta a troca das ordens dos trabalhos, não será necessário definir nei_1 nem nei_2

Desta vez a viabilidade é garantida, pois apenas os trabalhos concluídos dentro do período definido são contabilizados. Para estes dois últimos modelos utilizou-se o mesmo algoritmo *greedy* para a descodificação da solução.

Continua a ser importante escolher corretamente uma boa combinação de níveis para as variáveis devido ao grande impacto da interação destas sobre a qualidade das soluções e do tempo computacional associado. Estes valores estão reportados na Tabela 5.3

Tabela 5.3: Níveis das variáveis dos modelos de *Simulated Annealing*

Modelo	$\chi(c_0)$	CP	L_k	α	P
M1 NNGV	0.4, 0.6, 0.8	10, 25, 50, 75	500, 1000, 2500	0.8, 0.9, 0.95, 0.975	5, 10, 30
M1 NGV	0.4, 0.6, 0.8	10, 25, 50, 75	500, 1000, 2500	0.8, 0.9, 0.95, 0.975	5, 10, 30
M1 GV	0.4, 0.6, 0.8	10, 25, 50, 75	50, 100, 250	0.8, 0.9, 0.95, 0.975	-
M2	0.4, 0.6, 0.8	10, 25, 50, 75	50, 100, 250	0.8, 0.9, 0.95, 0.975	5, 10, 30
M3	0.4, 0.6, 0.8	10, 25, 50, 75	50, 100, 250	0.8, 0.9, 0.95, 0.975	-

Este novo problema apresenta de novo dificuldade em encontra os melhores valores para

Bibliografia

[1] M. Meskarpour Amiri, M. Kazemian, Z. Motaghed e Z. Abdi, «Systematic Review of Factors Determining Health Care Expenditures,» *Health Policy and Technology*, vol. 10, n.º 2, p. 100 498, 2021-06, ISSN: 2211-8837. DOI: 10.1016/j.hlpt.2021.01.004. acedido em 2025-05-16 (ver p. 1).

[2] E. R. de Saúde, *ERS - Tempos Máximos de Resposta Garantidos (TMRG)*, https://www.ers.pt/pt/utentes/p frequentes/faq/tempos-maximos-de-resposta-garantidos-tmrg/. acedido em 2025-05-16 (ver p. 1).

[3] A. Almén, «Trends in Diagnostic Nuclear Medicine in Sweden (2008–2023): Utilisation, Radiation Dose, and Methodological Insights,» *EJNMMI Physics*, vol. 12, n.º 1, p. 32, 2025-04, ISSN: 2197-7364. DOI: 10.1186/s40658-025-00747-2. acedido em 2025-05-21 (ver p. 1).

[4] S. Dauzère-Pérès, J. Ding, L. Shen e K. Tamssaouet, «The Flexible Job Shop Scheduling Problem: A Review,» *European Journal of Operational Research*, vol. 314, n.º 2, pp. 409–432, 2024-04, Fala sobre as restrições extras que pode haver, e tem bastantes artigos citados com no-wait e blocking-, ISSN: 0377-2217. DOI: 10.1016/j.ejor.2023.05.017. acedido em 2025-04-06 (ver pp. 3, 8).

[5] P. Brucker e R. Schlie, «Job-shop scheduling with multi-purpose machines,» *Computing*, vol. 45, n.º 4, pp. 369–375, 1990-12, ISSN: 1436-5057. DOI: 10.1007/BF02238804. acedido em 2025-05-24 (ver p. 3).

[6] P. Brandimarte, «Routing and scheduling in a flexible job shop by tabu search,» *Annals of Operations Research*, vol. 41, n.º 3, pp. 157–183, 1993-09, ISSN: 1572-9338. DOI: 10.1007/BF02023073. acedido em 2025-04-09 (ver p. 3).

[7] P. Brucker e J. Neyer, «Tabu-search for the multi-mode job-shop problem,» *Operations-Research-Spektrum*, vol. 20, n.º 1, pp. 21–28, 1998-03, ISSN: 1436-6304. DOI: 10.1007/BF01545525. acedido em 2025-05-26 (ver p. 3).

[8] S. Dauzère-Pérès, W. Roux e J. B. Lasserre, «Multi-Resource Shop Scheduling with Resource Flexibility,» *European Journal of Operational Research*, vol. 107, n.º 2, pp. 289–305, 1998-06, ISSN: 0377-2217. DOI: 10.1016/S0377-2217(97)00341-X. acedido em 2025-04-08 (ver p. 3).

[9] J. R. Callahan, *The Nothing Hot Delay Problem in the Production of Steel.* [Toronto], 1971, Open Library ID: OL19839735M (ver p. 4).

[10] B. Jiang, Y. Ma, L. Chen, B. Huang, Y. Huang e L. Guan, «A Review on Intelligent Scheduling and Optimization for Flexible Job Shop,» *International Journal of Control, Automation and Systems*, vol. 21, n.º 10, pp. 3127–3150, 2023-10, ISSN: 2005-4092. DOI: 10.1007/s12555-023-0578-1. acedido em 2025-04-10 (ver pp. 4, 7).

[11] A. E. Ezugwu et al., «Metaheuristics: A comprehensive overview and classification along with bibliometric analysis,» *Artificial Intelligence Review*, vol. 54, n.º 6, pp. 4237–4316, 2021-08, ISSN: 1573-7462. DOI: 10.1007/s10462-020-09952-0. acedido em 2025-07-10 (ver p. 5).

[12] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller e E. Teller, «Equation of State Calculations by Fast Computing Machines,» *The Journal of Chemical Physics*, vol. 21, n.º 6, pp. 1087–1092, 1953-06, ISSN: 0021-9606. DOI: 10.1063/1.1699114. acedido em 2025-05-28 (ver p. 6).

[13] H. Xiong, S. Shi, D. Ren e J. Hu, «A Survey of Job Shop Scheduling Problem: The Types and Models,» *Computers & Operations Research*, vol. 142, p. 105 731, 2022-06, ISSN: 0305-0548. DOI: 10.1016/j.cor.2022.105731. acedido em 2025-04-16 (ver p. 7).

[14] W.-Y. Ku e J. C. Beck, «Mixed Integer Programming Models for Job Shop Scheduling: A Computational Analysis,» *Computers & Operations Research*, vol. 73, pp. 165–173, 2016-09, ISSN: 0305-0548. DOI: 10.1016/j.cor.2016.04.006. acedido em 2025-07-07 (ver p. 7).

[15] P. Fattahi, M. Saidi Mehrabad e F. Jolai, «Mathematical modeling and heuristic approaches to flexible job shop scheduling problems,» *Journal of Intelligent Manufacturing*, vol. 18, n.º 3, pp. 331–342, 2007-06, ISSN: 1572-8145. DOI: 10.1007/s10845-007-0026-8. acedido em 2025-07-15 (ver p. 8).

[16] C. Özgüven, L. Özbakır e Y. Yavuz, «Mathematical Models for Job-Shop Scheduling Problems with Routing and Process Plan Flexibility,» *Applied Mathematical Modelling*, vol. 34, n.º 6, pp. 1539–1548, 2010-06, ISSN: 0307-904X. DOI: 10.1016/j.apm.2009.09.002. acedido em 2025-05-26 (ver p. 8).

[17] *(PDF) A competitive iterative procedure using a time-indexed model for solving flexible job shop scheduling problems*, https://www.researchgate.net/publication/311966224_A_competitive_iterat
indexed_model_for_solving_flexible_job_shop_scheduling_problems. acedido em 2025-07-14 (ver p. 8).

[18] H. Samarghandi, «Solving the No-Wait Job Shop Scheduling Problem with Due Date Constraints: A Problem Transformation Approach,» *Computers & Industrial Engineering*, vol. 136, pp. 635–662, 2019-10, ISSN: 0360-8352. DOI: 10.1016/j.cie.2019.07.054. acedido em 2025-07-15 (ver p. 8).

[19] R. Behmanesh e I. Rahimi, «Improved ant colony optimization for multi-resource job shop scheduling: A special case of transportation,» *Economic Computation and Economic Cybernetics Studies and Research*, vol. 55, n.º 4, pp. 277–294, 2021, ISSN: 0424-267X. DOI: 10.24818/18423264/55.4.21.18 (ver p. 8).

[20] M. Nawaz, E. E. Enscore e I. Ham, «A Heuristic Algorithm for the *M*-Machine, *n*-Job Flow-Shop Sequencing Problem,» *Omega*, vol. 11, n.º 1, pp. 91–95, 1983-01, ISSN: 0305-0483. DOI: 10.1016/0305-0483(83)90088-9. acedido em 2025-08-14 (ver p. 8).

[21] K.-C. Ying e S.-W. Lin, «Minimizing Total Completion Time in the No-Wait Jobshop Scheduling Problem Using a Backtracking Metaheuristic,» *Computers & Industrial Engineering*, vol. 169, p. 108 238, 2022-07, ISSN: 0360-8352. DOI: 10.1016/j.cie.2022.108238. acedido em 2025-08-14 (ver pp. 8, 9).

[22] M. Xu, S. Zhang e G. Deng, «No-Wait Job Shop Scheduling Using a Population-Based Iterated Greedy Algorithm,» *Algorithms*, vol. 14, n.º 5, p. 145, 2021-05, ISSN: 1999-4893. DOI: 10.3390/a14050145. acedido em 2025-08-14 (ver p. 8).

[23] D. Delahaye, S. Chaimatanan e M. Mongeau, «Simulated Annealing: From Basics to Applications,» em *Handbook of Metaheuristics*, M. Gendreau e J.-Y. Potvin, eds., Cham: Springer International Publishing, 2019, pp. 1–35, ISBN: 978-3-319-91086-4. DOI: 10.1007/978-3-319-91086-4_1. acedido em 2025-05-26 (ver p. 8).

[24] W. Xia e Z. Wu, «An Effective Hybrid Optimization Approach for Multi-Objective Flexible Job-Shop Scheduling Problems,» *Computers & Industrial Engineering*, vol. 48, n.º 2, pp. 409–425, 2005-03, ISSN: 0360-8352. DOI: 10.1016/j.cie.2005.01.018. acedido em 2025-08-14 (ver p. 9).

[25] S. Jonathan, C. E. Nugraheni e L. Abednego, «A Whale Optimization Algorithm Based Solver for No-wait Flexible Flow Shop Scheduling Problems,» em *2024 Ninth International Conference on Informatics and Computing (ICIC)*, 2024-10, pp. 1–6. DOI: 10.1109/ICIC64337.2024.10956970. acedido em 2025-08-14 (ver p. 9).

[26] Q. Perrachon, A.-L. Olteanu, M. Sevaux, S. Fréchengues e J.-F. Kerviche, «Industrial Multi-Resource Flexible Job Shop Scheduling with Partially Necessary Resources,» *European Journal of Operational Research*, vol. 320, n.º 2, pp. 309–327, 2025-01, ISSN: 0377-2217. DOI: 10.1016/j.ejor.2024.07.023. acedido em 2025-08-13 (ver p. 9).

[27] A. Caumond, P. Lacomme e N. Tchernev, «A Memetic Algorithm for the Job-Shop with Time-Lags,» *Computers & Operations Research*, Part Special Issue: Includes Selected Papers Presented at the ECCO'04 European Conference on Combinatorial Optimization, vol. 35, n.º 7, pp. 2331–2356, 2008-07, ISSN: 0305-0548. DOI: 10.1016/j.cor.2006.11.007. acedido em 2025-08-14 (ver p. 9).

[28] C. J. Schuster, «No-wait Job Shop Scheduling: Tabu Search and Complexity of Subproblems,» *Mathematical Methods of Operations Research*, vol. 63, n.º 3, pp. 473–491, 2006-07, ISSN: 1432-5217. DOI: 10.1007/s00186-005-0056-y. acedido em 2025-08-15 (ver p. 16).

[29] J. M. Framinan e C. Schuster, «An Enhanced Timetabling Procedure for the No-Wait Job Shop Problem: A Complete Local Search Approach,» *Computers & Operations Research*, vol. 33, n.º 5, pp. 1200–1213, 2006-05, ISSN: 0305-0548. DOI: 10.1016/j.cor.2004.09.009. acedido em 2025-08-15 (ver p. 16).

[30] J. Zhu, X. Li e Q. Wang, «Complete Local Search with Limited Memory Algorithm for No-Wait Job Shops to Minimize Makespan,» *European Journal of Operational Research*, vol. 198, n.º 2, pp. 378–386, 2009-10, ISSN: 0377-2217. DOI: 10.1016/j.ejor.2008.09.015. acedido em 2025-08-14 (ver p. 16).