DESENVOLVIMENTO DE UMA FERRAMENTA DE PRÉ-SELECÇÃO DE RECURSOS PARA EMPRESAS ÁGEIS/VIRTUAIS

Álvaro José Zenha Coelho e Castro

Mestrado em Engenharia Electrotécnica e de Computadores
Área de Especialização de Sistemas e Planeamento Industrial
Departamento de Engenharia Electrotécnica
Instituto Superior de Engenharia do Porto
2010

Candidato: Álvaro José Zenha Coelho e Castro, Nº 1040122, 1040122@isep.ipp.pt
Orientação científica: Paulo Ávila, psa@isep.ipp.pt
Co-orientação científica: António Pires, ant@isep.ipp.pt

Mestrado em Engenharia Electrotécnica e de Computadores
Área de Especialização de Sistemas e Planeamento Industrial
Departamento de Engenharia Electrotécnica
Instituto Superior de Engenharia do Porto
5 de Novembro de 2010

Agradecimentos

Uma dissertação, apesar do processo solitário a que qualquer investigador está destinado, reúne contributos de várias pessoas, o que torna por vezes esta parte dos agradecimentos uma tarefa difícil uma vez que nos podemos esquecer de mencionar alguém que contribui de forma importante para o trabalho.

Começo por apresentar os meus agradecimentos aos Engenheiros Paulo Ávila e António Pires pela atenção, tempo dispendido e ajudas prestadas nas diferentes fases desta dissertação e que contribuíram para o meu desenvolvimento profissional e pessoal.

Não podia deixar de agradecer ao Professor Lino Costa da Universidade do Minho pela sua disponibilidade de ajuda em momentos menos inspirados da minha parte no que concerne à parte inicial do ciclo de programação.

Por último, mas não menos importante, agradeço aos meus pais, demais família e amigos próximos o apoio prestado e a compreensão pelo pouco tempo dispendido com eles durante todo o tempo em que estive absorto na dissertação.

A todos, um muito obrigado.

Resumo

Actualmente o ciclo de vida de um produto ou serviço termina mais cedo em comparação com o que acontecia há alguns anos atrás. Isso deve-se à constante evolução da concorrência empresarial, aliado ao facto das necessidades dos consumidores também se alterarem num curto espaço de tempo. Assim, e para dar resposta a estas solicitações do mercado, as empresas como modo de sobrevivência, têm de se adaptar, tornando-se mais flexíveis ou agilizando o seu sistema de produção. No entanto, por vezes isto acarreta elevados custos que as empresas não são capazes de suportar. Surge assim o conceito de Empresa Ágil/Virtual (E A/V) que consiste numa colaboração/cooperação em rede entre diversas empresas de forma à criação de um produto ou serviço tendo em conta uma janela temporal de oportunidade.

O processo de criação/desenvolvimento de uma E A/V não é simples. Devido ao elevado número de potenciais recursos disponíveis é importante fazer uma selecção daqueles que mais garantias dão em termos de sucesso nessa cooperação e que mais valor trazem para a E A/V. Existem diversos projectos de E A/V com diferentes abordagens ao processo de selecção. O projecto *Business Model: Virtual Enterprise Architecture Reference Model* (BM_VEARM) incorpora um modelo de selecção de recursos que será tratado nesta dissertação. Tendo em consideração os aspectos desse modelo, vai ser desenvolvida, como objectivo principal desta dissertação, uma ferramenta de pré-selecção de recursos em MATLAB e sua integração com outra ferramenta para a selecção final. O resultado será a ferramenta Selec_E A/V_V2 com as seguintes funcionalidades: Pré-selecção de recursos sem integração da análise de valor (AV), com integração de AV e selecção final de recursos atendendo às duas situações referidas. Esta ferramenta trará suporte ao desenvolvimento do modelo de selecção BM_VEARM e possibilitará a validação da importância da AV no processo de pré-selecção de recursos, processo este que é tido como objecto de análise numa tese de doutoramento em progresso.

Palavras-Chave

Empresa Ágil/Virtual, selecção de recursos, pré-selecção de recursos, Análise do Valor, MATLAB

Abstract

Nowadays the life cycle of a product or service finishes earlier in comparison to what we can observe in the last pasting years. This is due to the constant evolution of business competition, coupled with the fact that the consumer's demands tend to change in a short period of time. To meet these market demands, the companies, in order to survive, have to adapt, becoming more flexible or finding ways to make their production system faster. However, sometimes this involves high costs that firms are not able to bear. In order to overcome those problems arises the concept of Agile/Virtual Enterprise (A/V E) which is a collaboration/networking between different companies in order to create a product or service within an opportunity time window.

The process of creation/development of an A/V E is not simple. Due to the large number of potential resources is important to make a selection of those who give more guarantees in terms of success in this cooperation and bring more value to A/V E. There are several projects of A/V E with different approaches to the selection process. The Business Model: Virtual Enterprise Architecture Reference Model (BM_VEARM) project incorporates a model of resource selection that will be addressed in this dissertation. Considering the aspects of the model, the main objective of this dissertation, is the development of a tool for pre-selection of resources in MATLAB, and its integration with another tool for the final selection. The result is the tool named Selec_E A/V_V2 with the following features: Pre-selection of resources without integration of the value analysis (VA), with integration of VA and final selection of resources given to those mentioned cases. This tool will support the development of BM_VEARM selection model and will permit the validation of the importance of value analysis (VA) in the resources pre-selection, a process that is being analyzed in a Ph.D. thesis in progress.

Keywords

Agile/Virtual Enterprise, resource selection, resource pre-selection, value analysis, MATLAB.

Índice

Índice de Figuras

Índice de Tabelas

Acrónimos

AV	–	Análise de Valor
BM_VEARM	–	*Business Model: Virtual Enterprise Architecture Reference Model*
CP	–	Custos de Processamento
CT	–	Custos de Transporte
DRP	–	Vector de matrizes que reúne a informação global de todo o PT em questão
Drp	–	Conjunto de recursos candidatos a cada tarefa
Drp_apr	–	Conjunto de recursos aprovados no 1º nível da pré-selecção
Drp_final	–	Conjunto de recursos aprovados nos 2 níveis da pré-selecção
E A/V	–	Empresa Ágil/Virtual
MR	–	Mercado de Recursos
PT	–	Plano de Tarefas

1. INTRODUÇÃO

Ao longo dos tempos tem-se verificado uma constante evolução por parte das empresas na sua politica organizacional. Esta evolução deve-se essencialmente à crescente concorrência empresarial e que, acompanhada por bruscas mudanças das necessidades dos consumidores, faz com que as empresas tenham de, rapidamente, modificar a sua estrutura organizacional de forma a dar resposta às tendências e solicitações do mercado, [3][5].

Em empresas de pequenas e médias dimensões, a exigência de grande flexibilidade e dinâmica para a resposta ao mercado é sinónimo de elevados custos (aquisição de novas tecnologias, desenvolvimento e produção de novos produtos) pondo por vezes em causa a sobrevivência da empresa [7][18]. Uma solução encontrada é a cooperação entre um variado número de empresas, cooperação essa que permite que as empresas se especializem/concentrem cada vez mais na sua actividade base, nas suas *core competencies*, estabelecendo ligações com entidades externas, através de *outsourcing* ou outras, para a produção ou fornecimento de um serviço.

Cresce assim o conceito de empresas ágeis, empresas que devido às constantes mudanças nos mercados têm de ter a capacidade de se adaptarem. Para essa adaptação ser possível, é necessário integrarem tecnologias de produção flexíveis, um grupo de trabalho competente e com os conhecimentos adequados, bem como uma gestão capaz de iniciar e incentivar ligações cooperativas entre outras empresas. Possuindo pelo menos estes factores, aliados a um elevado grau de confiança entre as empresas que cooperam entre si e à partilha de conhecimento e informação, a empresa será capaz de responder às exigências do mercado,

tendo a capacidade de introdução de novos produtos, de acordo com a constante evolução das exigências dos clientes [3].

Com o conceito de empresa ágil está também associado o conceito de empresa virtual. Uma empresa virtual, no contexto da produção, pode ser entendida como um conjunto de diversas empresas ágeis a interagirem entre elas por um determinado intervalo de tempo com a intenção de atingirem determinados objectivos bem definidos. Cada empresa participa com a sua competência principal, *core competence*, criando alianças estratégicas de forma a poderem entrar/explorar novos mercados (ver Figura 1). O conceito de uma empresa virtual pode ser resumido como sendo uma empresa temporária, em que no final do tempo necessário para concretizar os objectivos propostos dá-se o seu término. Para isso, as responsabilidades, competências e lucros são estipulados e acordados previamente para que quando a missão/projecto terminar esta possa ser desfeita rapidamente e sem grandes burocracias ou custos associados.

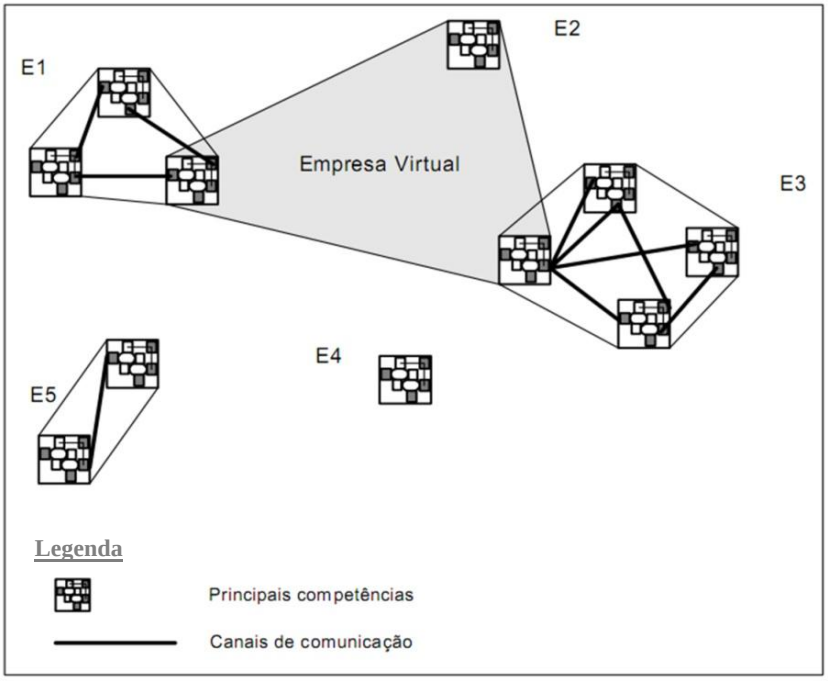

Figura 1 Exemplo de uma Empresa Virtual [15]

1.1. CONTEXTUALIZAÇÃO

Este trabalho está inserido no âmbito da selecção de sistemas de recursos para a criação de uma Empresa Ágil/Virtual (E A/V). Para a criação de uma E A/V é necessário proceder a uma escolha das melhores empresas/recursos para a tarefa/produto em questão. Tendo isso em atenção é necessário identificar e avaliar uma série de parâmetros que, de acordo com algum tipo de modelo de selecção de recursos nos permita identificar o(s) recurso(s) mais vantajosos para a E A/V em causa.

A fase de pré-selecção é extremamente importante, proporcionando uma posterior selecção dos recursos que possuem os requisitos necessários para a tarefa em questão, o que agiliza a escolha do sistema de recursos, reduzindo a quantidade de informação a tratar por excluir recursos que não respeitam os requisitos mínimos.

1.2. OBJECTIVOS

O principal objectivo desta dissertação, tal como o próprio título refere, é o desenvolvimento de uma ferramenta de pré-selecção de recursos que visa tornar o processo de pré-selecção, e consequentemente a selecção de recursos, mais rápido e eficiente. Para a criação dessa ferramenta é necessário:

- Compreender a pré-selecção de recursos para E A/V;
- Desenvolver o algoritmo de pré-selecção em pseudo-código;
- Implementar o algoritmo de pré-selecção em código MATLAB e integrar com a selecção final;
- Simular e validar o código criado;

1.3. METODOLOGIA DE INVESTIGAÇÃO

Existem diversas metodologias de investigação e abordagens distintas para a análise de um determinado problema. No que diz respeito a esta dissertação, utilizamos a metodologia de investigação designada por aplicada. Este tipo de metodologia visa encontrar uma solução para um problema relacionado com sociedades ou organizações industriais, servindo também como forma de validação de uma teoria [8]. Esta metodologia aplicada é aquela que melhor se enquadra com os objectivos da dissertação.

1.4. ORGANIZAÇÃO DO RELATÓRIO

No capítulo 1 são explicados de forma simples os conceitos associados às E A/V e é efectuada uma contextualização do âmbito em que esta dissertação se vai centrar. No capítulo seguinte, 2, vão ser explicados os processos da selecção de recursos, tendo em conta o modelo BM_VEARM. No capítulo 3 é explicitado o problema em análise, assim como a criação de um determinado pseudo-código para linguagem MATLAB tendo como finalidade a criação de uma ferramenta de pré-selecção para integração com a selecção final de recursos. No capítulo 4 é feita uma pequena introdução à ferramenta que vai servir de base para o desenvolvimento do programa proposto, MATLAB, e será feita uma explicação mais pormenorizada do que cada um dos módulos do código constituintes do programa irá fazer. No capítulo seguinte, 5, será feita a validação da ferramenta criada, ou seja, será representada uma simulação, que será explicada passo a passo, mostrando todas as etapas do programa e comprovando que os cálculos efectuados pela ferramenta estão correctos. O capítulo 6 corresponde às conclusões finais da dissertação.

2. PROCESSO DA SELECÇÃO DE RECURSOS PARA E A/V SEGUNDO MODELO BM_VEARM

De acordo com o que foi referido na introdução desta dissertação, a agilidade e flexibilidade das empresas em se adaptarem às constantes solicitações do mercado são factores extremamente importantes para o seu sucesso. Para a selecção dos recursos candidatos a um determinado plano de tarefas (PT) de uma E A/V existem diversos modelos e métodos de selecção. Contudo, para determinadas situações os modelos e métodos existentes demonstram não serem os melhores, nomeadamente pela falta de flexibilidade para corresponder aos requisitos e especificidades da nova E A/V. Não é portanto viável que para o inicio de cada E A/V seja criado novo modelo de selecção. Tendo isso em consideração, o nosso modelo referencial, BM_VEARM, proposto por Putnik[16], assenta numa estrutura hierárquica multi-nível de processos, que visa satisfazer as propriedades básicas de uma empresa virtual, nomeadamente a integrabilidade, distributividade, agilidade e virtualidade, usando para isso três ferramentas: o Mercado de Recursos (MR), o *Broker* e a Virtualidade [17]. O MR é uma ferramenta que à primeira

impressão pode parecer um simples "catálogo" ou base de dados com informação organizada com vista a facilitar a escolha de recursos. Contudo, é uma ferramenta que permite agilizar o processo de criação de uma E A/V na medida em que actua em diversos pontos críticos como o processo de transacção (que engloba o tempo gasto em pesquisas, contratualização, monitorização e execução/integração dos recursos) e na questão dos recursos disponibilizados serem dignos de confiança no que toca à partilha de informação (protecção de propriedade intelectual, partilha entre terceiros). Além do mais disponibiliza também ferramentas de ajuda à tomada de decisão, bem como uma mediação entre a procura e oferta de recursos para integração numa E A/V. A segunda ferramenta, o *Broker*, tal como o MR é uma entidade externa independente da E A/V e tem duas funções essenciais. A primeira consiste em ser o agente responsável pela agilização e dinâmica na reconfiguração da organização controlando tudo o que diz respeito aos recursos, desde a sua selecção, integração, reconfiguração, monitorização e análise, o que implica um profundo conhecimento na pesquisa e integração de recursos e na gestão da reconfiguração das E A/V. A segunda função está relacionada com o facto de o *Broker* ser o agente da virtualidade. Isto implica que o *Broker* seja o intermediário entre o cliente (promotor da E A/V) e o recurso, não havendo contacto directo entre estas duas entidades. Isto permite ao *Broker*, fazer uma reestruturação da organização durante o tempo de operação, caso verifique que um dos recursos não está a cumprir com os objectivos definidos, tendo no entanto o requisito de o cliente não ser prejudicado de alguma forma, ver Figura 2.

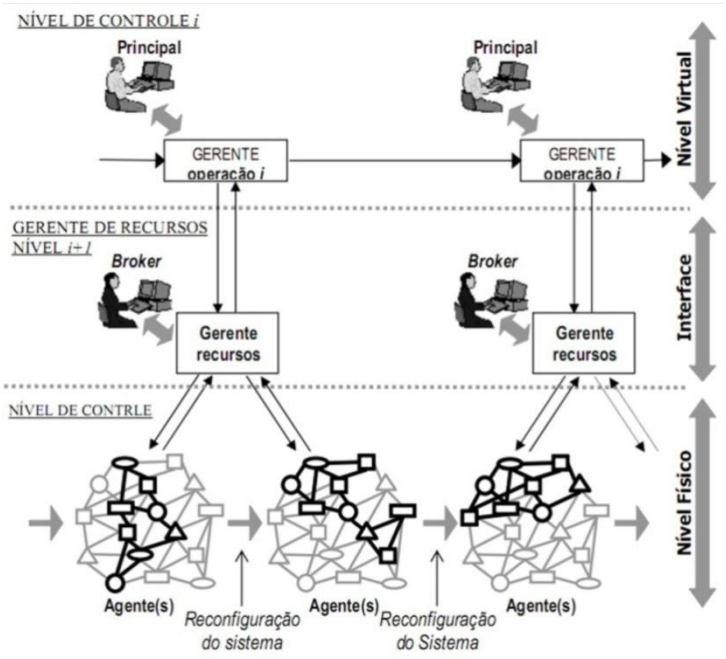

Figura 2 Esquema de operação da E A/V [15]

Desta forma, a actuação do *Broker* em tempo real, juntamente com a estrutura hierarquizada do cliente/*Broker*/recursos constituem a terceira ferramenta do modelo, a Virtualização. A introdução de um *Broker*/negociador vai-se traduzir numa forma mais eficaz e eficiente de selecção de recursos para este tipo de empresas [2]. No modelo em questão está prevista a actuação do *Broker* no processo de selecção de recursos, embora a opção da sua utilização seja sempre da E A/V. Na Figura 3 temos uma apresentação do processo de selecção proposto.

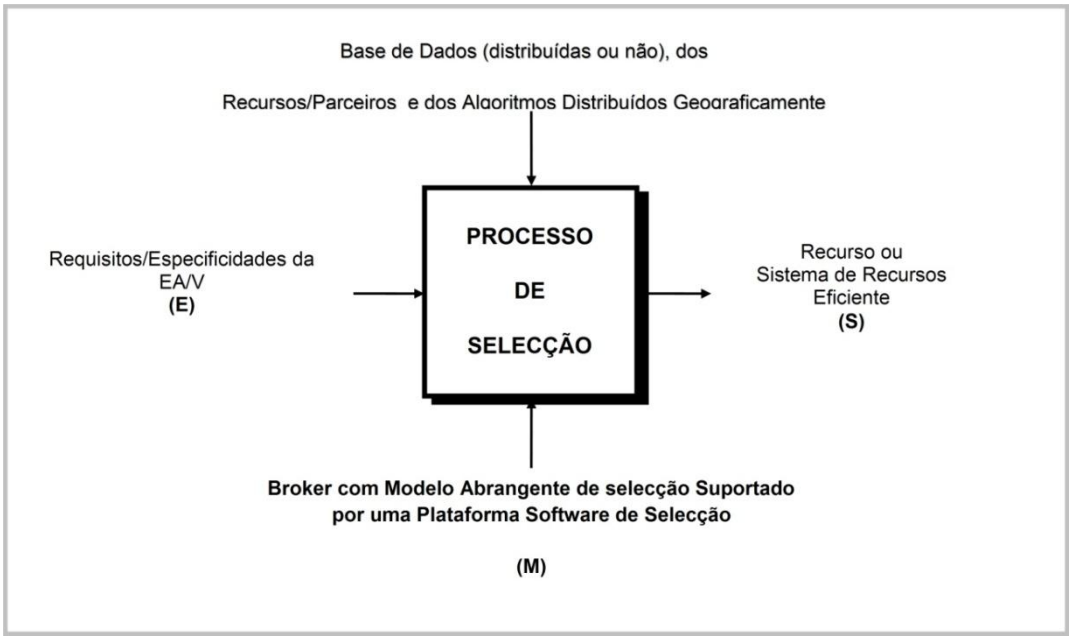

Figura 3 Uma especificação do processo de selecção proposto [2]

O modelo de selecção proposto divide-se em duas fases principais, pré-selecção e selecção de recursos (Figura 4). A rapidez e qualidade dos recursos seleccionados em ambas as fases são de extrema importância para a criação da E A/V. Como estas duas fases são sequenciais, primeiro a pré-selecção e só depois a selecção final, os resultados da primeira fase vão influenciar e condicionar os resultados da segunda, especialmente no que concerne ao n.º de recursos pré-seleccionados, que vai afectar a complexidade e o tempo de análise dos dados [2].

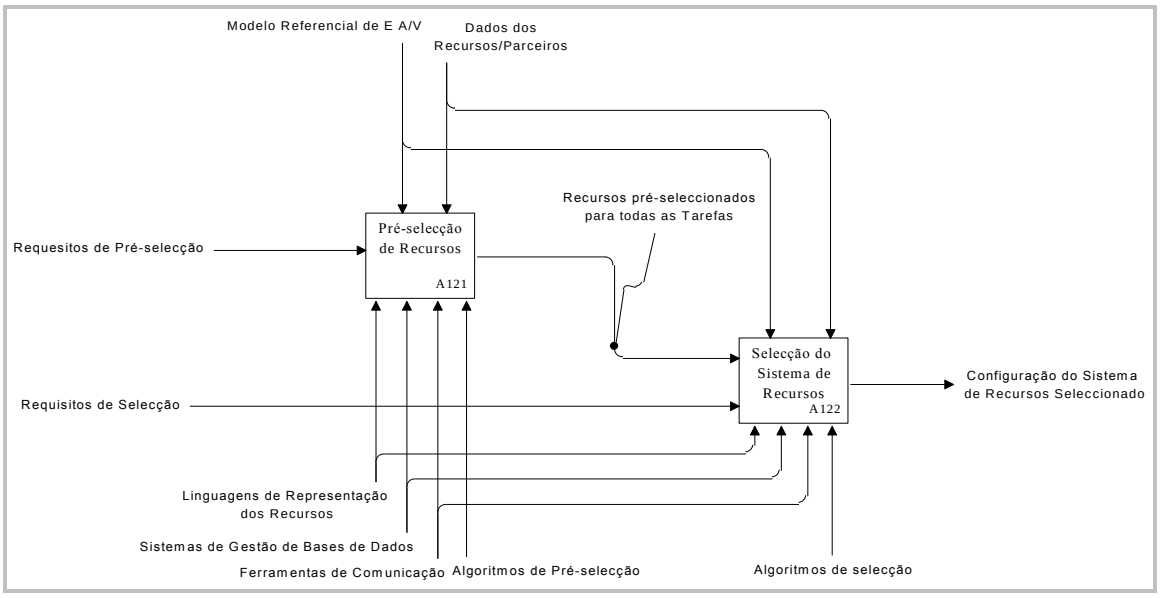

Figura 4 Fases da selecção de recursos [2]

2.1. FASE DA PRÉ-SELECÇÃO

Na figura seguinte aparece representado o processo de pré-selecção de recursos

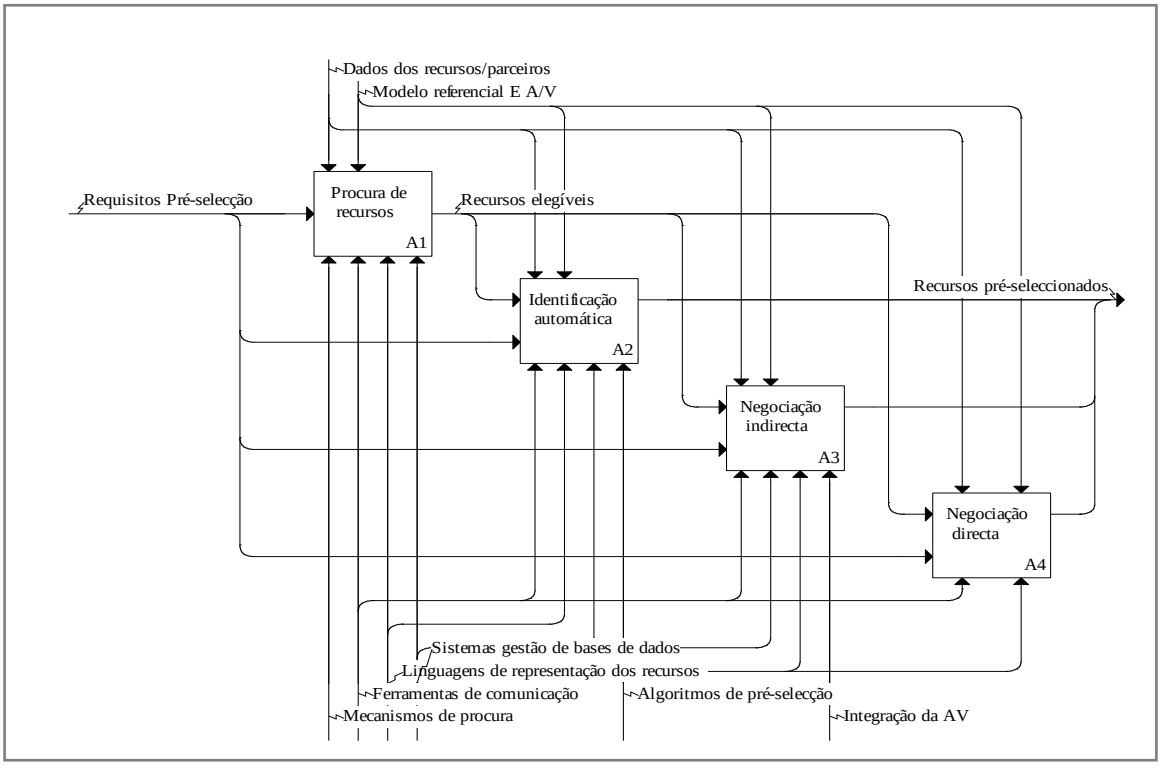

Figura 5 Representação do processo de pré-selecção de recursos com integração AV [14]

O processo da pré-selecção contempla duas fases principais, a procura de recursos e a sua pré-selecção, seja ela por identificação automática, negociação indirecta ou negociação directa.

2.1.1. FASES DA PRÉ-SELECÇÃO DE RECURSOS [2]

Antes de podermos iniciar qualquer selecção de recursos é necessário reunirmos um conjunto de recursos para que os possamos avaliar e consequentemente considerá-los como aptos ou não para a tarefa em questão.

Procura de recursos

A procura de recursos é feita com o objectivo de reunir uma lista de vários recursos que preencham da melhor forma possível os parâmetros da pré-selecção. O resultado desta pesquisa é um conjunto de recursos potencialmente interessados na participação da execução de tarefas numa E A/V, ou seja, recursos elegíveis, os quais passarão à fase de pré-selecção propriamente dita.

Identificação automática

No processo de identificação automática, pressupõe-se que os parâmetros de pré-selecção dos recursos elegíveis estejam disponíveis nalgum tipo de base de dados em suporte informático, ou outro. Para que a comparação dos diversos parâmetros seja efectuada de uma forma rápida é desejável que estes estejam organizados num formato *standard*, mediante o tipo de tarefa/produto.

Negociação indirecta

A Negociação indirecta pode ser considerada como um tipo de leilão. As tarefas são dadas a conhecer aos potenciais recursos candidatos do mundo empresarial e os recursos que estejam interessados fazem as suas respectivas propostas. Seguidamente essas propostas são analisadas e aceites se corresponderem aos requisitos necessários. Este processo de selecção acarreta um tempo de execução superior ao referido anteriormente, sendo no entanto bastante utilizado até ao momento devido ao facto de a identificação automática não ser exequível em grande parte dos casos.

Negociação directa

Negociação directa é o processo de selecção mais demorado pois implica uma negociação directa (por parte da empresa ou mediante um *Broker*/negociador) com os recursos elegíveis relativamente às suas propostas face aos requisitos da pré-selecção.

No contexto desta dissertação, vamos focar-nos na pré-selecção por meio de negociação indirecta com integração de AV por ser aquele com maior flexibilidade e que melhor se ajusta aos objectivos pretendidos.

2.1.2. REQUISITOS DA FASE DE PRÉ-SELECÇÃO DE RECURSOS [2]

Na fase de pré-selecção de recursos são considerados alguns requisitos tais como: Produto/Tarefa; Projecto do Produto/Tarefa; Processo de Fabrico; Planeamento da Produção e Outros. De uma forma resumida estes requisitos consistem em:

Produto / Tarefa

Consiste nas especificações funcionais do produto associadas a cada tarefa, como por exemplo, especificações para o projecto do produto, especificações do plano de controlo da qualidade e características dos materiais.

Projecto do Produto / Tarefa

Está associado ao projecto e modelação do produto / tarefa e respeita por exemplo a: cálculos, modelos CAD do produto, de conjunto e/ou de pormenor.

Processo de Fabrico

Diz respeito às operações e sua sequência para a tarefa em causa, especificação de cada operação com a identificação dos processos respectivos, tipos de máquinas e ferramentas para cada operação, ferramentas de controlo, dimensões operacionais e tolerâncias para as operações.

Planeamento da Produção

Consiste no planeamento e programação da produção, por exemplo nas quantidades necessárias entre intervalos de tempo (datas de início e conclusão das tarefas).

Outros

Foram englobados nesta categoria, outros requisitos considerados relevantes nos modelos analisados, tais como, sistemas de gestão da qualidade, localização geográfica, situação financeira, aspectos culturais, factores organizacionais, histórico de colaborações e outros. Também as tecnologias de informação e sua compatibilidade são fundamentais para o projecto e integração de uma E A/V, uma vez que a sua estrutura organizacional requer um elevado nível de cooperação e coordenação inter-organizacional. De notar que esta funcionalidade não foi objecto de análise individualizada nos modelos existentes por entendermos que é uma área fora do âmbito do presente trabalho e que está subjacente ao próprio conceito de E A/V. Estes requisitos pertencem ao que consideramos o nível 1 da pré-selecção.

O nível 2 da pré-selecção consiste na integração da AV tendo em conta a ponderação dos Sistemas considerados nessa avaliação. Cada um dos Sistemas em análise (ver Tabela 1) terá um conjunto de requisitos com determinada classificação, que depois de ponderados e avaliados nos permitirá obter um determinado valor para esse sistema. Após a ponderação individual dos requisitos associados a cada sistema, haverá uma ponderação global dos 3 sistemas, obtendo-se assim o valor do recurso candidato.

Tabela 1 Níveis dos Sistemas de Funcionalidades do Modelo Proposto com AV [1]

Nível	Sistemas de Funcionalidades
1	**Produto/Tarefa** **Projecto e Modelação do Produto/Tarefa** **Processo de Fabrico / Tarefa** **Planeamento e Programação da Produção**
2	**Sistema da Qualidade** **Sistema Financeiro** **Sistema de Sinergias**

De seguida será efectuada uma breve descrição do que consiste cada um destes 3 Sistemas que serão considerados para a integração da AV [1]

Sistema da Qualidade

Consiste nos sistemas de gestão da qualidade, garantias, nível de serviço, qualidade centrada no cliente e conceitos TQM.

Sistema Financeiro

Engloba os rácios económicos e financeiros, indicadores da criação de valor, estabilidade financeira, contratualização e integração em mercados de capitais.

Sistema de Sinergias

Foram incluídos nesta categoria factores considerados relevantes, tais como, potencial de parceria, localização, aspectos culturais e estratégicos e relações inter-organizacionais.

2.2. FASE DA SELECÇÃO FINAL

Tendo os resultados da pré-selecção irá ser feita a selecção final do sistema de recursos para o PT em questão. Nesta fase, os resultados serão imediatamente integrados no bloco de Selecção Final do Sistema (A1123 da Figura 6) na medida em que os dois processos anteriores ainda estão em vias de serem implementados.

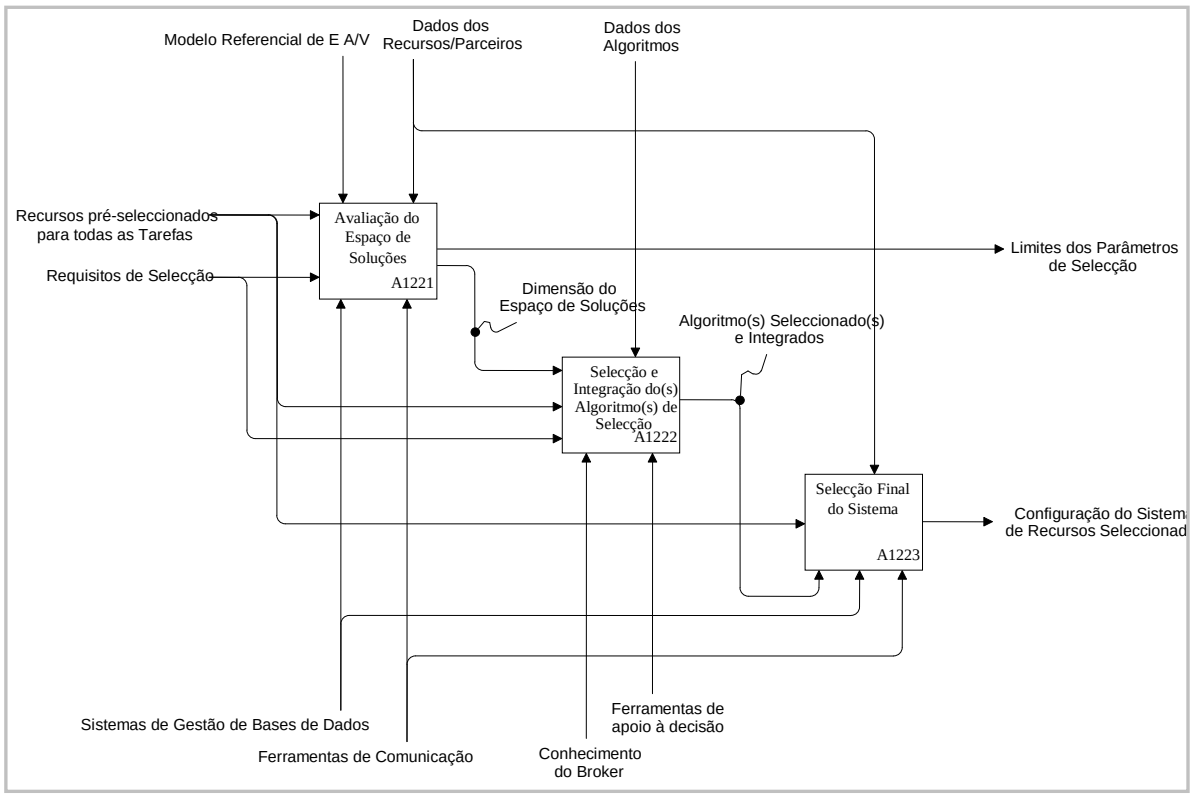

Figura 6 Representação da selecção do sistema de recursos [2]

Apesar de neste momento os dois primeiros processos ainda não estarem implementados funcionalmente no modelo, são de grande importância para uma melhor selecção final do sistema de recursos e irão ser explicados de forma sucinta de seguida. Para a consulta na íntegra ver capítulo 5 de Ávila [2].

Avaliação do espaço de soluções

Este processo consiste na análise da viabilidade de a E A/V avançar com o projecto ou não. Recebendo os requisitos de selecção e os recursos pré-seleccionados como *inputs*, é possível quantificar, de forma grosseira, os limites do desempenho dos sistemas de recursos. Mediante os limites quantificados, a E A/V tem a opção de cancelar o avanço do projecto ou considerar um novo ciclo de pré-selecção considerando novos recursos.

Selecção e integração dos algoritmos de selecção

Neste processo, e tal como o nome indica, vai ser feita a selecção do(s) algoritmo(s) a serem utilizados para a determinação do sistema de recursos para cada PT. A utilização de um algoritmo pode ser suficiente para um determinado PT mas pode não reunir as condições necessárias para um PT que reúna outro tipo de requisitos. De notar também que o problema de selecção de recursos pode ser multi-critério, sendo por isso necessário a escolha de mais do que um algoritmo de selecção.

Selecção Final do Sistema

Nesta fase, os eventuais algoritmos que teriam sido seleccionados no processo anterior serão postos a correr de forma a se obter o sistema de recursos para o PT. Neste momento, e como os dois processos anteriores não estão ainda implementados, será apenas utilizado o algoritmo de selecção final do tipo *Branch&Bound*.

3. EXPLICITAÇÃO DO PROBLEMA

O principal âmbito deste trabalho consiste na criação de uma ferramenta de entrada de dados (*inputs*) para posterior pré-selecção de recursos. Esta ferramenta deve ser passível de ser integrada com um algoritmo de selecção final do tipo *Branch&Bound* já desenvolvido [1].

Sendo assim, e depois de analisada a questão da pré-selecção de recursos, é criado o pseudo-código apresentado de seguida, para que depois se possa passar à criação do código na linguagem MATLAB, seguido da integração com o algoritmo de selecção final.

Este pseudo-código está dividido por módulos para ser mais perceptível, ou seja, para que seja possível entender de uma forma mais simples as funcionalidades de cada um dos módulos. No capítulo seguinte, cada módulo vai ser explicado de uma forma mais detalhada.

Módulo Base

Inicio

Carrega inputs

Pede os mínimos aceitáveis para os níveis 1 e 2 da pré-selecção (*Sma_1* e *Sma_2*)

Pede o n.º de Plano de Tarefas a analisar (*nPT*)

Para o n.º de *nPT*

Pergunta o n.º de tarefas correspondente a cada PT (*nTare*)

Para o n.º de *nPT*

Para o n.º de *nTare*

Pergunta o n.º de recursos candidatos (*Xc*)

Pede os valores de ponderação para os requisitos de cada sistema, e ponderação final (*pondq*, *pondf*, *ponds*, *PondF*)

Confirma a validade dos valores de ponderação (soma == 100%)

Transforma os dados de ponderação introduzidos

Cria os vectores de matrizes (*valor* e *valor*) que vão armazenar os 2 resultados da selecção final (sem e com AV)

Para cada PT

Chama a função (módulo) da pré-selecção (**pré_Selec**)

Encontra o *k_min1* para o nível 1 do algoritmo de pré-selecção

Chama a função (módulo) da selecção final (**Selec_BB**)

Cria o vector *locali* com as localizações dos recursos seleccionados e respectivos CT (valor=1 no vector *x1*)

Para de 1 a *nTare*

Vai copiar o n.º do recurso (identificação do recursos) seleccionado para a 1ª coluna da matriz da 1ª célula do vector de matrizes *valor*

Vai copiar a classificação ponderada do recurso seleccionado para a 2ª coluna da matriz da 1ª célula do vector de matrizes *valor*

Vai copiar o custo de processamento do recurso seleccionado para a 3ª coluna da matriz da 1ª célula do vector de matrizes *valor*

Para de 1 a *nTare*

Vai somar o valor da classificação ponderada dos recursos seleccionados e armazena na 2ª célula do vector de matrizes *valor*

Armazena o vector dos custos de processamento (*CP*) na 3ª célula do vector de matrizes *valor*

Armazena o vector dos custos de transporte (*CT*) na 4ª célula do vector de matrizes *valor*

Armazena os custos totais do sistema (*fx1*) na 5ª célula do vector de matrizes *valor*

Chama a função (módulo) da apresentação de resultados da selecção (**Apr_res_selec**)

Encontra o *k_min2* para o nível 2 do algoritmo de pré-selecção

Se *k_min2*==0

Mostra mensagem e muda o nome das variáveis *DRP* e *x1*

senão

Altera o vector dos *CT* e *CP* para serem feitos cálculos c/ AV

Chama a função (módulo) da selecção final (**Selec_BB**)

Cria o vector *locali2* com as localizações dos recursos seleccionados e respectivos CT (valor=1 no vector *x2*)

Para de 1 a *nTare*

Vai copiar o n.º do recurso seleccionado para a 1ª coluna da matriz da 1ª célula do vector de matrizes *valor2*

Vai copiar a classificação ponderada do recurso seleccionado para a 2ª coluna da matriz da 1ª célula do vector de matrizes *valor2*

Vai copiar o custo de processamento do recurso seleccionado para a 3ª coluna da matriz da 1ª célula do vector de matrizes *valor2*

Para de 1 a *nTare*

Vai somar o valor da classificação ponderada dos recursos seleccionados e armazena na 2ª célula do vector de matrizes *valor2*

Armazena o vector dos custos de processamento (*CP2*) na 3ª célula do vector de matrizes *valor2*

Armazena o vector dos custos de transporte (*CT2*) na 4ª célula do vector de matrizes *valor2*

Armazena os custos totais do sistema (*fx2*) na 5ª célula do vector de matrizes *valor2*

Chama a função (módulo) da apresentação de resultados da selecção (**Apr_res_selec**)

Apresenta o tempo total da simulação

Muda o nome das variáveis *x1*, *x2* e *DRP* para que os dados não sejam reescritos em cada ciclo

FIM

Módulo pré_Selec

Inicio

Cria um vector de matrizes (*DRP*) de 6 linhas por *nTare* colunas

Para de 1 a *nTare*

Cria uma matriz (*Drp*) de *Xc* linhas por 21 colunas

Para de 1 a *Xc* (cada recurso)

1ª coluna da matriz *Drp* é o n.º do recurso (identificação do recurso)

Da coluna 2 a 5 é preenchida usando uma distribuição binomial com probabilidade de sucesso de 90%

As colunas 6 a 20 são preenchidas usando uma distribuição normal de média 6 e desvio padrão de 2

A coluna 21 é preenchida usando uma distribuição normal de média 6 e desvio padrão de 2

É verificada a validade dos dados da coluna 6 a 20 (se estão entre o intervalo [0 10])

Avaliação de nível 1

Cria a matriz *Drp_apr* com *Xc* linhas por 17 colunas

Para de 1 a *Xc* (cada recurso)

Copia o n.º do recurso para a 1ª coluna da matriz *Drp_apr*

Se o recurso atingir os mínimos do nível 1 (colunas 2 a 5 == *Sma_1*)

Copia as colunas correspondentes às classificações dos sistemas e CP para a matriz *Drp_apr*

senão

Elimina a linha do recurso na matriz *Drp_apr* e continua o ciclo

Se não existirem recursos no final deste ciclo

Imprime mensagem de erro

Avaliação de nível 2

Inicia a matriz *Drp_final* com o n.º de linhas de *Drp_apr* e 6 colunas

Para cada recurso da matriz *Drp_apr*

Copia o n.º do recurso da matriz *Drp_apr* para a matriz *Drp_final*

Calcula a classificação ponderada do sistema da qualidade e guarda essa classificação na 2ª coluna da matriz *Drp_final*

Calcula a classificação ponderada do sistema financeiro e guarda essa classificação na 3ª coluna da matriz *Drp_final*

Calcula a classificação ponderada do sistema das sinergias e guarda essa classificação na 4ª coluna da matriz *Drp_final*

Calcula a classificação ponderada dos 3 sistemas e guarda essa classificação na 5ª coluna da matriz *Drp_final*

Copia os custos de processamento (CP) associados a cada recurso para a 6ª coluna da matriz *Drp_final*

Guarda a matriz *Drp_final* na 3ª célula, coluna correspondente à tarefa, do vector de matrizes *DRP* ordenada em ordem decrescente do valor da 5ª coluna (classificação ponderada dos 3 sistemas, C.Pond) para que possa ser reutilizada sem perder dados

São contabilizados os recursos que têm classificação positiva em cada um dos sistemas e também no conjunto dos 3

Para o n.º de recursos de *Drp_final*

Se o recurso não atingir os mínimos do nível 2 (colunas 2 a 4 <*Sma_2*)

Preenche a linha referente ao recurso com zeros

Todas as linhas com zeros são eliminadas

São apresentados os resultados da pré-selecção

A matriz *Drp* é armazenada na 1ª célula, coluna correspondente à tarefa, do vector de matrizes *DRP*

A matriz *Drp_apr* é armazenada na 2ª célula, coluna correspondente à tarefa, do vector de matrizes *DRP*

A matriz *Drp_final* é armazenada na 5ª célula, coluna correspondente à tarefa, do vector de matrizes *DRP*

Considera-se o mesmo n.º de recursos aprovados no 1º nível para todas as tarefas (*kmin_1*) dentro do mesmo PT

Verifica a dimensão da matriz final de cada tarefa e armazena o valor mínimo (*DIM_min*)

Copia a 3ª célula de cada coluna do *DRP* para a 4ª célula

Para cada tarefa

Preenche a linha com zeros desde a posição *DIM_min*+1 até à dimensão da matriz

Todas as linhas com zeros são eliminadas

Criar vector de *CP* de todos os recursos que passam nível 1

Para cada tarefa

Cria o vector *Cp* que contém todos os valores da 6ª coluna da matriz da 4ª célula do *DRP*

Concatena o vector *Cp* ao vector *CP* que contém os custos de processamento dos recursos de todas as tarefas

Considera-se o mesmo n.º de recursos aprovados no 2º nível para todas as tarefas (*k_min2*) dentro do mesmo PT

Verifica a dimensão da matriz final de cada tarefa e armazena o valor mínimo (*DIM2_min*)

Copia a 5ª célula de cada coluna do *DRP* para a 6ª célula

Para cada tarefa

Preenche a linha com zeros desde a posição *DIM2_min*+1 até à dimensão da matriz

Todas as linhas com zeros são eliminadas

FIM

Módulo Apr_res_selec

Inicio

Se *Flag_ni2*==0

Apresenta cabeçalho referente à selecção sem análise de valor

senão

Apresenta cabeçalho referente à selecção com análise de valor

Apresenta o sistema de recursos seleccionado

Concatena a letra "R" com o n.º da tarefa e o n.º do recurso seleccionado

Apresenta os custos do sistema (*fx*) armazenados na 5ª célula do vector de matrizes *valor*

Faz a soma da 3ª coluna da matriz da 1ª célula do vector de matrizes *valor* para apresentar os CP totais

De 1 a *nTare*

Concatena ["CP R" + n.º da tarefa + "," + n.º do recurso seleccionado + "=" + CP associado ao recurso escolhido]. Para apresentar CP de cada recurso escolhido

Subtrai os CP totais aos custos do sistema para apresentar os CT totais

De 1 a *nTare*-1

Concatena [“CT ” + n.º da tarefa + “->” + n.º da tarefa seguinte + “=” + CT associado]. Para apresentar CT de cada recurso escolhido

Apresenta valor do sistema armazenado na 2ª célula do vector de matrizes *valor*

De 1 a *nTare*

Concatena [“Valor R” + n.º da tarefa + “,” + n.º do recurso seleccionado + “=” + valor associado ao recurso escolhido]. Para apresentar o valor de cada recurso escolhido (2ª coluna da matriz da 1ª célula de *valor*).

Apresenta o n.º de recursos com classificação negativa (para os resultados sem AV)

Apresenta os resultados da *exitflag*

Apresenta o n.º de iterações e o tempo de simulação

FIM

Feita a conversão deste pseudo-código para linguagem MATLAB é necessário fazer a integração com o algoritmo de selecção final já existente.

A primeira vez que o programa efectua a chamada do módulo de selecção final não houve qualquer problema de funcionamento. No entanto, quando se tentou chamar novamente a selecção final para o cálculo do sistema de recursos tendo em conta a AV, constatou-se que existia um problema de dimensões dos vectores de CT e CP. Relativamente ao necessário redimensionamento do vector dos CP foi criado novo vector, *CP2*, sendo esta uma questão de mais fácil resolução pois estes acompanham sempre os respectivos recursos nas diversas fases da pré-selecção. No que diz respeito ao redimensionamento dos CT o grau de dificuldade aumenta. Com a enorme diferença de dimensões do vector *CT* a ter em conta entre o processo de selecção sem AV e com AV é necessário analisar as ligações entre os recursos que foram eliminados do 1º para o 2º nível da pré-selecção. (Figura 20 e Figura 21). Para que a coerência entre os CT entre os recursos aprovados das diversas tarefas se mantenha é necessário reestruturar o vector dos CT original em forma de matriz e proceder à eliminação dessas ligações que deixaram de existir. De seguida vai ser apresentado de forma sucinta o pseudo-código associado a esse redimensionamento.

De 1 a *nTare*-1

    Redimensionar parte do vector *CT* numa matriz de dimensões *k_min1* por *k_min1*

    Para cada par de tarefas (1-2, 2-3, 3-4,…)

        Para cada tarefa

            Comparar os recursos que passam os diferentes níveis da pré-selecção.

            Descobrir as posições em que os recursos que foram eliminados estavam na matriz dos que passam o 1º nível

            Preencher com zeros as linhas/colunas, da matriz com os CT redimensionados, nas posições em que se encontravam esses recursos

Copiar os dados que sobraram para o novo vector *CT2*

No final da integração, o software criado deve ser capaz de realizar aquilo para que foi proposto, ou seja:

- Permitir a introdução dos *inputs* relativos ao PT a analisar;
- Preencher automaticamente, segundo as distribuições definidas, todos os parâmetros relativos aos recursos de cada tarefa;
- Efectuar uma correcta pré-selecção de nível 1 dos recursos;
- Calcular correctamente o valor de cada recurso mediante as ponderações introduzidas pelo utilizador;
- Efectuar uma correcta pré-selecção de nível 2 tendo em conta a AV dos recursos;
- Integrar com algoritmo de selecção final do tipo *Branch&Bound*;
- Armazenar os valores para que depois da simulação possam ser analisados caso seja pretendido;
- Apresentar os resultados globais da pré-selecção e da selecção final;
- Funcionar como demonstrador de modo a permitir a validação de modelo de pré-selecção com integração de AV.

4. IMPLEMENTAÇÃO NO MATLAB DE UM ALGORITMO DE PRÉ-SELECÇÃO

Neste capítulo vai ser feita uma breve introdução à ferramenta que vai servir de base para o desenvolvimento do programa proposto no contexto desta dissertação, bem como uma explicação mais pormenorizada do que cada módulo do código da ferramenta vai efectuar.

4.1. A FERRAMENTA MATLAB

O MATLAB, abreviação de MATrix LABoratory, é um programa vocacionado para o cálculo numérico. A primeira versão foi escrita no final da década de 70 por Cleve Moler, utilizando a linguagem Fortran, com o objectivo de ser uma ferramenta de ajuda ao ensino. Devido à sua crescente popularização, quer na área do ensino como da investigação, foi reescrito e comercializado em linguagem C. Neste momento é uma ferramenta usada em diversas áreas de engenharia, ciências matemáticas e mesmo na indústria como ferramenta de pesquisa, desenvolvimento e análise [6][9].

Apesar de o programa organizar os dados em vectores e matrizes é um programa com uma rápida curva de aprendizagem [10]. Escrito numa linguagem de alto nível e, de certa forma, mais fácil de utilizar do que a linguagem C, proporciona um processo de declaração das variáveis simplificado, não sendo necessário definir qual o seu tipo (inteiro, char). Permite também a implementação de uma área de trabalho agradável para o utilizador e a disponibilização de uma interface interactiva que permite uma fácil identificação e correcção de erros que possam ocorrer.

A questão da portabilidade dos programas desenvolvidos em diferentes sistemas operativos está ultrapassada visto que o MATLAB tem suporte em diversas distribuições quer do Windows (9x/NT/2000/XP/Vista/7) quer de sistemas UNIX. Desta forma, os dados gerados por um programa escrito em Windows XP podem ser analisados/reutilizados usando um sistema operativo diferente, mediante as preferências dos utilizadores.

Outro ponto a favor está relacionado com a disponibilização de um amplo conjunto de funções predefinidas e ferramentas que facilitam a programação. As funções, que podem ir de uma simples organização dos dados por ordem crescente a um tratamento estatístico dos mesmos, são soluções testadas e disponibilizadas ao utilizador evitando que este tenha de fazer as suas próprias rotinas e facilitando assim o seu trabalho. Para além deste tipo de funções existem, como já foi referido, várias ferramentas que se aplicam em áreas específicas, mediante as necessidades do utilizador. Neste caso, o utilizador pode comprar um pacote de ferramentas para resolução de problemas relacionados com processamento de sinais, comunicações, braços robóticos, entre outras [4]. Existe ainda outra ferramenta que permite o desenvolvimento de uma interface gráfica para o utilizador. Isto permite que um programa algo complexo possa ser utilizado por uma pessoa menos experiente e de uma forma mais agradável.

Devido ao facto do módulo de selecção final já ter sido desenvolvido nesta plataforma, aliado às vantagens acima mencionadas, decidiu-se implementar a ferramenta de pré-selecção também em MATLAB. Isto vai tornar possível uma integração com a fase final da selecção de uma forma muito mais expedita do que caso se optasse por algum outro tipo de plataforma de desenvolvimento.

4.2. EXPLICAÇÃO DAS VÁRIAS FASES DO CÓDIGO

A ferramenta de selecção de recursos, cujo código se apresenta no Anexo B, é constituída no total por quatro módulos (Selec_E A/V_V2, pre_Selec, Selec_BB e Apr_res_selec). Nesta fase vai ser feita a descrição do que cada módulo faz e no capítulo seguinte vai ser apresentado um exemplo concreto da aplicação. Por uma questão de facilitar a explicação vamo-nos referir ao módulo Selec_E A/V_V2 como módulo Base.

4.2.1. MÓDULO BASE

O fluxograma da Figura 7 representa de forma sucinta o que o módulo vai efectuar.

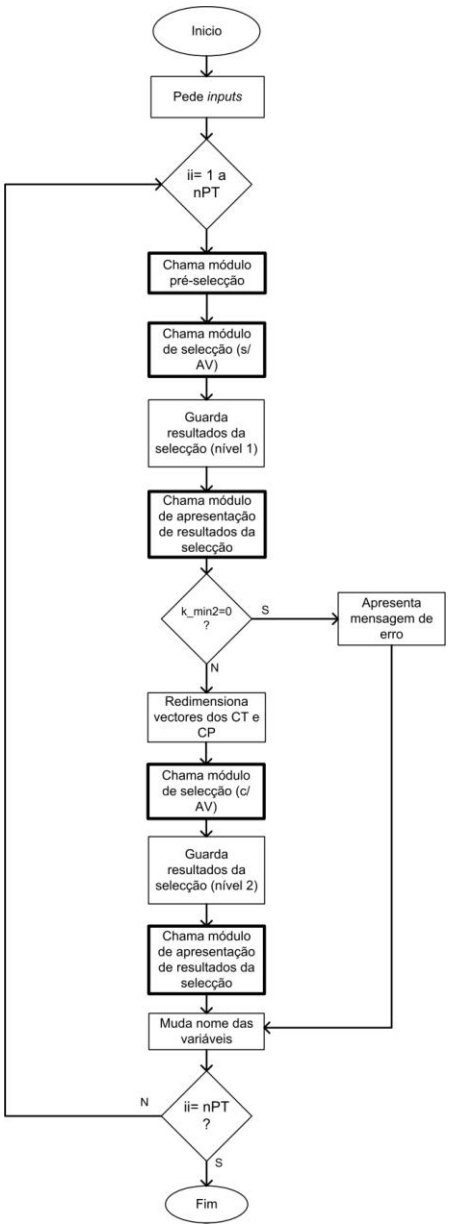

Figura 7 Fluxograma do módulo Base

O módulo Base é o módulo que inicia o programa de selecção de fornecedores/recursos. Neste módulo vão ser introduzidos os *inputs* necessários do programa. Estes *inputs* referem-se aos mínimos admissíveis para cada um dos níveis da pré-selecção (*Sma_1* e *Sma_2*), número de PT a analisar (*nPT*), o número de tarefas correspondentes a cada PT (*nTare*) e o número de recursos candidatos a cada tarefa (*Xc*). Também nesta fase inicial são introduzidos os valores de ponderação dos requisitos associados a cada um dos sistemas, Qualidade, Financeiro, Sinergias (*pondq*, *pondf*, *ponds* respectivamente) e os valores para a ponderação final dos 3 sistemas (*PondF*). Após a introdução dos *inputs* necessários o programa entra em modo autónomo, não necessitando de mais nenhum tipo de manipulação por parte do utilizador.

Desta forma, após a introdução de todos os *inputs*, o programa vai fazer os cálculos necessários para cada PT. Na primeira fase destes cálculos começa-se por "chamar" a função/módulo pre_Selec (que será explicada pormenorizadamente mais à frente) que irá retornar como parâmetros de saída o vector de matrizes *DRP* de dimensões (6 por *nTare*) e o vector dos custos de processamento (*CP*) referentes a cada recurso.

Vai ser verificada a dimensão de uma das matrizes do vector *DRP* (a correspondente à 4ª linha e 1ª coluna). O valor dessa dimensão (*k_min1*) e o vector dos *CP* serão enviados como parâmetro de entrada para o módulo Selec_BB (do qual será dada mais à frente uma pequena explicação pois foi já desenvolvido [1]) que irá calcular o sistema de recursos sem ter em conta a AV. Esta função/módulo é responsável pela geração do vector dos custos de transporte (*CT*) e pela selecção dos recursos utilizando o algoritmo *Branch&Bound* como algoritmo de optimização. Esta função vai devolver um vector com os recursos seleccionados pelo algoritmo de optimização e com os CT associados (as localizações relativas dos recursos e dos CT associados são armazenados num vector de nome *locali*), os custos totais do sistema (*fx1*) e mais um conjunto de variáveis de controlo.

Os recursos escolhidos para cada tarefa bem como a sua classificação ponderada e os seus CP, serão armazenados na 1ª célula do vector de matrizes de nome *valor*. Na 2ª célula será armazenado o valor total do sistema, e na 3ª, 4ª e 5ª células serão armazenados os vectores dos *CP*, *CT* e os custos totais do sistema respectivamente.

De seguida é chamada a função/módulo Apr_res_selec para apresentação dos resultados da selecção sem AV. Esta função será também explicada mais pormenorizadamente ao longo do documento.

Após esta primeira fase de apresentação de resultados vão ser verificadas as condições para que se possa proceder a uma selecção de recursos tendo em conta a AV. Para tal vai ser verificada a dimensão da última matriz da primeira coluna do vector de matrizes *DRP*. Se a dimensão dessa matriz for superior a 0 (zero), *k_min2*, significa que existem recursos que passaram o 2º nível da pré-selecção e, como tal, poderá ser feita uma selecção final de recursos usando o algoritmo de optimização.

No entanto, e que para tal seja possível, é necessário proceder a um redimensionamento do vector *CT* previamente retornado pela função Selec_BB. Este redimensionamento é necessário uma vez que os CT foram gerados tendo em conta um determinado número de recursos, número esse que é diferente para esta selecção com a AV. É necessário descobrir os CT associados aos recursos que foram eliminados no 2º nível do algoritmo de pré-selecção. Assim, e de uma forma sucinta, são criadas matrizes de *k_min1*k_min1* elementos com os valores dos CT. As colunas e linhas dessas matrizes são preenchidas com 0 (zeros) caso o recurso correspondente tenha sido eliminado na 2ª fase do processo de pré-selecção. Os valores que se mantiverem são os referentes aos recursos que se mantiveram classificados, formando assim o novo vector dos CT (*CT2*) (este passo aparece melhor explicitado aquando da validação da ferramenta tendo em conta um exemplo simples).

Da mesma forma que o vector dos CT teve de ser redimensionado, o mesmo terá de acontecer ao vector *CP*, embora este processo se revele de forma mais simples uma vez que os CP dos recursos que foram aprovados na 2ª fase do processo de pré-selecção e que vão agora entrar para o cálculo da selecção final estão armazenados no vector de matrizes *DRP*, sendo apenas necessário percorrer a última coluna das matrizes referentes à 6ª (última) linha do *DRP*.

Uma vez que ambos os vectores, *CT* e *CP*, foram redimensionados, a função/módulo Selec_BB é chamada novamente, sendo estes vectores enviados como parâmetros de entrada.

Daqui para a frente o processo repete-se. A função Selec_BB vai retornar os recursos seleccionados, desta vez tendo em conta a AV, sendo o vector de matrizes *valor2* quem vai armazenar os parâmetros apontados anteriormente para o vector de matrizes *valor*.

Ainda neste módulo, é apresentado o tempo total da simulação, obtido a partir de um dos *outputs* da função Selec_BB.

4.2.2. MÓDULO/FUNÇÃO PRE_SELEC

Tal como no módulo Base, este módulo também pode ser representado na forma do fluxograma da Figura 8.

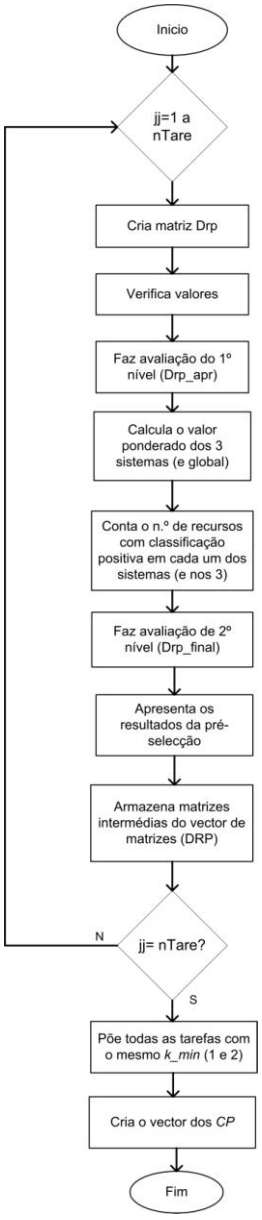

Figura 8 Fluxograma do módulo de pré-selecção

Neste módulo é feita, numa primeira fase, a geração dos valores dos recursos candidatos a cada tarefa. Enquanto no módulo Base, que trabalhava PT a PT, neste módulo vamos trabalhar tarefa a tarefa. Sendo assim, a primeira coisa a fazer é criar o vector de matrizes *DRP* (6 por *nTare*) onde vão ser armazenadas as matrizes de recursos que vão sendo criadas no desenrolar do processo de pré-selecção e que mais tarde vai ser retornado para o módulo Base.

Como este módulo/função vai ser chamado a partir do módulo Base, vai ter alguns parâmetros de entrada, nomeadamente o número de tarefas referente ao PT em questão (*nTare*), o número de recursos candidatos a cada tarefa (*Xc*), os vectores dos mínimos admissíveis para cada um dos níveis da pré-selecção (*Sma_1* e *Sma_2*), os factores de ponderação e o número do PT com que estamos a trabalhar.

Sendo assim, para cada tarefa vai ser criada uma matriz (de nome *Drp*) com as dimensões de *Xc* por 21. A primeira coluna vai ser referente ao n.º de identificação do recurso. As colunas 2 a 5 (referentes aos requisitos de 1º nível da pré-selecção) vão ser preenchidas utilizando uma distribuição binomial com percentagem de sucesso de 90%. Isto significa que nos 4 valores a serem gerados há no mínimo uma probabilidade de 90% de obtermos o valor 1 (um). Esta função é do tipo booleana, isto é, o valor 1 (um) significa aprovação e o valor 0 (zero) significa rejeição. Foi tida em conta esta percentagem para que nos casos em que temos poucos recursos candidatos (10 ou 15) ainda haja um n.º considerável de recursos (*k_min*) para a próxima fase do algoritmo de pré-selecção para podermos tirar conclusões acerca do desempenho do algoritmo. É também expectável que os *Xc*, respondam ao pedido de propostas somente se cumprirem estes requisitos básicos de nível 1, em todo o caso consideramos ainda então essa margem de 10% de *Xc* a rejeitar nesta fase. As colunas 6 a 20 vão ser preenchidas utilizando um distribuição normal de média 6 e desvio padrão de 2. Os valores destas colunas respeitam ao conjunto de requisitos dos 3 sistemas considerados (Qualidade, Financeiros e Sinergias). Na geração destes valores foi considerada a escala geralmente utilizada na análise do valor (0 a 10) [13]. Neste caso, foi considerada a média de 6 porque, também no caso de poucos recursos candidatos, seria expectável que a quantidade de recursos que ultrapassaria o 2º nível da pré-selecção (AV) fosse nulo na maioria das simulações. Desta forma garantimos em termos de simulações que teremos sempre pelo menos um candidato pré-seleccionado em cada tarefa. Também é expectável, que os recursos candidatos, tendo conhecimento dos requisitos associados as

estes 3 sistemas só respondam à proposta caso satisfaçam minimamente estes requisitos. A coluna 21, correspondente aos CP, vai ser também preenchida tendo em conta uma distribuição normal de média 6 e desvio padrão de 2. Foi considerada uma distribuição normal pois é isso que se verifica na realidade. Quando se fazem pesquisas de mercado para um mesmo produto/tarefa o custo centra-se num determinado valor, havendo como é óbvio pequenas oscilações. O valor da média (6) foi utilizado para manter a conformidade com a escala utilizada na geração dos restantes critérios de valor.

Após o preenchimento desta matriz inicial é feita uma verificação de que os valores dos requisitos dos sistemas supramencionados não ultrapassam a escala considerada (0-10).

Passando agora ao nível 1 da pré-selecção, vai ser criada uma nova matriz, de nome *Drp_apr* (matriz dos recursos aprovados no 1º nível), com as dimensões de *Xc* linhas por 17 colunas. Percorrendo a matriz inicial linha a linha vão ser efectuados os seguintes passos:

- O n.º de identificação do recurso é copiado para a 1ª coluna da matriz *Drp_apr*;
- Os requisitos do 1º nível da pré-selecção vão ser comparados com os mínimos admissíveis (*Sma_1*). Caso se verifiquem as condições mínimas, os valores referentes aos requisitos dos 3 sistemas e CP (colunas 6 a 21) são copiados para a matriz *Drp_apr* (colunas 2 a 17). Se os mínimos não se verificarem, a linha correspondente ao recurso que está a ser analisado é eliminado na matriz *Drp_apr*. Desta forma a matriz fica o mais reduzida possível, libertando memória no sistema.

No final, se a dimensão da matriz *Drp_apr* for nula, significa que nenhum recurso passou a avaliação de 1º nível.

Após esta primeira avaliação dos recursos, vai-se proceder à AV dos recursos, ou seja, vai ser feita a ponderação das classificações do conjunto dos requisitos para cada um dos sistemas considerados. Nesta fase irá ser criada uma nova matriz de nome *Drp_final* com o mesmo n.º de linhas de *Drp_apr* mas apenas com 6 colunas. A primeira coluna continua a ser referente à identificação do recurso, as colunas 2 a 4 correspondem à classificação ponderada de cada um dos sistemas, Qualidade, Financeiro e Sinergias respectivamente, a coluna 5 corresponde à classificação ponderada dos 3 sistemas, e a coluna 6 é a que armazena os CP do recurso.

Após estes cálculos, ao ser armazenada na 3ª linha (coluna correspondente ao n.º da tarefa em análise) do vector de matrizes *DRP*, as linhas desta matriz vão ser ordenadas por ordem decrescente do valor da 5ª coluna, correspondente ao valor da ponderação dos 3 sistemas, para que os recursos com melhor classificação, i.e. maior valor, fiquem nas posições cimeiras da matriz. Desta forma é possível continuar a trabalhar e a efectuar eventuais alterações na matriz *Drp_final*, não havendo a necessidade de criar mais matrizes auxiliares para optimização do programa e ficando sempre com estes dados disponíveis para consulta e verificação do correcto funcionamento do programa.

Para uma questão meramente informativa para o utilizador, esta matriz vai ser percorrida linha a linha para serem contabilizados os recursos com classificação positiva em cada um dos sistemas (valor igual ou superior a 5), bem como o n.º de recursos que tem classificação positiva nos 3 sistemas.

De seguida, passamos ao nível 2 do algoritmo de pré-selecção. Tal como já foi referido, é necessário que os recursos atinjam determinados níveis mínimos nas suas classificações para que possam passar à fase seguinte. No caso do nível 2 deste algoritmo, tendo em conta a situação definida para efeitos de simulação, foi considerada condição necessária de que os recursos tenham classificação positiva (superior ou igual a 5) nos 3 sistemas. De forma idêntica ao que se realizou para os requisitos de nível 1 também neste caso a matriz *Drp_final* vai ser percorrida linha a linha sendo que os valores das colunas 2 a 4, correspondentes às classificações em cada sistema, vão ser comparados com os mínimos admissíveis do nível 2 (*Sma_2*). Se se verificar a condição de que o valor obtido num sistema de determinado recurso não corresponde ao mínimo admissível, a linha correspondente a esse recurso vai ser preenchida com 0 (zeros). No final, todas as linhas preenchidas com zeros são eliminadas e a matriz é ordenada por ordem decrescente do valor da 5ª coluna.

Os resultados desta pré-selecção são apresentados ao utilizador tarefa a tarefa, isto é, de cada vez que este processo descrito é realizado são apresentados os resultados referentes à tarefa em questão. No final de cada tarefa as matrizes com que fomos trabalhando ao longo de todo este processo vão ser armazenadas no já referido vector de matrizes *DRP*. Na 1ª linha é armazenada a matriz inicial, *Drp*, na 2ª linha é armazenada a matriz *Drp_apr* e na 5ª linha é armazenada a matriz *Drp_final*. De relembrar que na 3ª linha já tinha sido

armazenada a matriz *Drp_final* antes de terem sido eliminados os recursos que não cumpriam os requisitos do 2º nível do algoritmo.

Fazendo uma breve resenha do que foi explicado até este momento (referente a este módulo), os valores dos recursos foram gerados automaticamente utilizando as distribuições referidas. Foi realizado o 1º nível do algoritmo de pré-selecção onde foram eliminados os recursos que não satisfaziam os requisitos básicos para a concretização da tarefa. De seguida foi feita a ponderação dos critérios dos 3 sistemas e uma classificação final ponderada de cada recurso. Foi então aplicado o 2º nível do algoritmo, eliminando os recursos que não possuíam (no modelo em questão) classificação positiva nos 3 sistemas de avaliação considerados. Foram armazenadas matrizes de dados nas linhas 1,2,3 e 5 do vector de matrizes *DRP* (uma coluna para cada tarefa).

Após todas as tarefas terem sido analisadas, e por ser um requisito do módulo de selecção final (Selec_BB), o n.º de recursos aprovados em cada nível tem de ser igual em todas as tarefas (*k_min1* ou *k_min2* para o nível 1 e nível 2 respectivamente) dentro do mesmo PT. Por esse motivo as linhas 4 e 6 do vector de matrizes *DRP* tinham sido deixadas em branco até ao momento. Para tal, as dimensões das matrizes referentes aos recursos que passam o 1º nível de cada tarefa (3ªlinha de cada coluna do vector de matrizes *DRP*) têm de ser analisadas. A de menor dimensão é que vai servir de referência para redimensionar todas as outras. Para este redimensionamento, as matrizes da 3ª linha do vector de matrizes *DRP* vão ser copiadas para a 4ª linha. De seguida, coluna a coluna (tarefa a tarefa), a matriz que foi copiada para a 4ª linha vai ser preenchida com 0 (zeros) a partir da dimensão de referência obtida anteriormente. As linhas com 0 (zeros) serão posteriormente eliminadas. O mesmo redimensionamento vai ser efectuado para os recursos que passam o 2º nível. Desta vez, as matrizes da linha 6 de *DRP* terão a mesma dimensão da menor matriz da linha 5, utilizando para isso o mesmo método referido há pouco. Esta situação vai ser explicitada no capítulo referente à validação da ferramenta, 5.

Este vector de matrizes *DRP* será um dos *outputs* deste módulo. O outro *output* será o vector com os CP dos diversos recursos que passaram o 1º nível do algoritmo. Para construir este vector é necessário percorrer a 4ª linha do vector *DRP*, coluna a coluna, e agrupar os elementos da última coluna das matrizes. Esse vector será o vector dos CP que depois será utilizado no módulo da selecção final.

4.2.3. MÓDULO SELEC_BB

Este módulo é o responsável pela selecção final dos recursos e vai ser chamado duas vezes. Da primeira vez vai fazer a análise dos recursos sem ter em conta a AV, recebendo apenas como parâmetros de entrada *nTare*, correspondente ao PT em análise, o n.º de recursos pré-seleccionados considerados para todas as tarefas (*k_min1*) e o vector dos CP associados aos recursos. Vai ser também responsável, da 1ª vez, pela criação do vector dos CT que, em conjunto com os CP, vão ser as condicionantes a ter em conta para a selecção dos recursos (o algoritmo de optimização utilizado visa a minimização dos custos). Da 2ª vez que é chamado, recebe também como parâmetro de entrada os novos vectores de CP (*CP2*) e CT (*CT2*) redimensionados tendo em consideração o n.º de recursos que passam o 2º nível da pré-selecção (*k_min2*). Desta vez, os resultados obtidos, já têm em conta a AV dos recursos.

Como outputs deste módulo temos, quer para a 1ª ou 2ª chamada, um vector *x* onde estão simbolizadas as posições dos recursos seleccionados e os CT respectivos, o valor do custo do sistema, *fx* (CP+CT), uma *exitflag*, que nos permite saber se a solução encontrada foi ou não óptima e se não foi, qual o motivo (o número de iterações foi excedido, não tem solução possível, o tempo de execução foi excedido, o número de *nodes* foi excedido, o número de iterações realizadas sobre o mesmo *node* foi excedido).

4.2.4. MÓDULO APR_RES_SELEC

Este módulo é responsável pela apresentação dos resultados da selecção, com e sem integração da AV.

Este módulo recebe vários parâmetros de entrada que vão ser necessários para a apresentação dos resultados obtidos pelo módulo de selecção final. Vai receber o vector de matrizes que armazena os resultados da selecção final, *valor*; o vector com as localizações dos recursos seleccionados e com os CT associados, *locali*; o n.º de tarefas, *nTare*, e o n.º de recursos candidatos às tarefas, *Xc*; o n.º do PT em análise (no caso de estarmos a simular para mais do que um); uma *flag* de controlo (*Flag_ni2*); os dois *outputs* de controlo retornados pelo módulo de selecção final e as dimensões das matrizes dos recursos que passam o 1º e o 2º nível do algoritmo de pré-selecção.

Quando o módulo é chamado, se a *Flag_ni2* ainda é 0 (zero) vai apresentar um conjunto de informações, género de cabeçalho. Esse conjunto de informações passa pelo n.º do PT em

análise, *nTare* e *Xc* (antes de qualquer pré-selecção), o n.º de recursos que passam o nível 1 da pré-selecção, *k_min1*, (tendo em conta a normalização do n.º de recursos igual para todas as tarefas) e a indicação de que os dados apresentados vão ser referentes à selecção de recursos sem ter em conta a AV. Ainda para o caso de a *Flag_ni2* ser 0 (zero), é apresentada uma informação, já fora do que é considerado cabeçalho, do n.º de recursos seleccionados mas que têm valor abaixo do mínimo admissível (considerado 5 para efeitos de simulação). Quando o módulo é chamado pela 2ª vez a *flag* de controlo, *Flag_ni2* será 1 (um). Deste modo, irão ser apresentados *Xc*, o n.º de recursos que passa o 1º e o 2º nível da pré-selecção (*k_min1* e *k_min2*) e a indicação de que os dados são referentes à selecção com AV.

A apresentação do cabeçalho é a única coisa que difere entre as análises (com ou sem AV), tudo o resto é similar. De cada vez que o módulo é chamado, apresenta, para além do respectivo cabeçalho, o sistema de recursos seleccionado e o custo total do sistema, fazendo a distinção entre CP e CT. Tanto nos CP como nos CT são também apresentadas a diversas parcelas que irão constituir os seus custos totais. É apresentado também o valor total do sistema e, tal como nos custos do sistema, é feita a apresentação individual do valor de cada um dos recursos seleccionados.

Para além da apresentação dos resultados da selecção (com ou sem AV) são também apresentados outros dados referentes à simulação propriamente dita. Um dos dois *outputs* de controlo do módulo de selecção é designado por *exitflag*. Este *output* é um valor devolvido pelo módulo de selecção e indica se a solução encontrada foi óptima ou se ocorreu algum tipo de problema para que essa solução óptima não fosse encontrada. O outro *output* é uma estrutura que engloba várias informações e é mesmo denominado por *ouput1* ou *output2* mediante o tipo de análise em questão (sem ou com AV). Neste caso, vão ser retiradas o n.º de iterações realizadas e o tempo que demorou a simulação. De notar que o tempo obtido vai variar entre os sistemas (as características do computador influenciam o tempo de processamento) em que as simulações forem realizadas. Como tal, o tempo utilizado para simulações com parâmetros de entrada idênticos, pode variar.

5. VALIDAÇÃO DA FERRAMENTA

Neste capítulo vamos efectuar uma simulação de forma a verificar se a ferramenta cumpre com os requisitos propostos, servindo também para melhor consolidar o que foi explicado no capítulo anterior. No entanto, é de salientar que o exemplo apresentado vai seguir a ordem do programa, ou seja, vai ser apresentada a introdução dos dados iniciais e será apresentado o funcionamento do módulo da pré-selecção, voltando ao módulo Base. De seguida será chamado o módulo da selecção final sem ter em conta a AV (do qual não será feita nenhuma explicação pormenorizada) e será explicado o armazenamento dos dados resultantes da selecção. Após a explicação da apresentação dos resultados será detalhado em pormenor o redimensionamento do vector *CT* e *CP*. É chamado novamente o módulo de selecção desta vez tendo em conta a AV dos recursos.

Inicio do programa

Os mínimos admissíveis para cada um dos níveis de pré-selecção (*Sma_1* e *Sma_2*) estão introduzidos por defeito, sendo no entanto possível alterar de forma simples o programa para que peça ao utilizador.

Inputs

São introduzidos os *inputs* necessários já referidos (representados na Figura 9), n.º de PT a analisar (*nPT*), o n.º de tarefas referente a cada plano (*nTare*), o n.º de recursos candidatos a cada tarefa do plano (*Xc*) e as ponderações para os sistemas tidos em consideração para a AV.

```
Número de plano de tarefas a analisar: 1

Para o plano de tarefas 1 indique número de tarefas
N.º de tarefas: 4

Plano de tarefas n.º 1
O número de recursos é igual para todas as tarefas? (s/n): s

Introduza o número de recursos candidatos às tarefas: 10

Introduza os valores de ponderação para os 3 sistemas em % (Se todos iguais prima enter):

Introduza os 5 valores de ponderação da qualidade em % (Se todos iguais prima enter):

Introduza os 5 valores de ponderação financeiros em % (Se todos iguais prima enter):

Introduza os 5 valores de ponderação das Sinergias em % (Se todos iguais prima enter):
```

Figura 9 Introdução de dados

Pré-selecção

De seguida está representado o código que chama o módulo da pré-selecção.

```
[DRP, CP]=pre_Selec(nTare(ii), Xc{ii}, Sma_1,
pondq, pondf, ponds, PondF, Sma_2,ii);
```

Como se pode verificar, são enviados como parâmetros de entrada o n.º de tarefas do plano, o n.º de recursos candidatos, o vector dos mínimos admissíveis para o nível 1, as ponderações dos 3 sistemas e ponderação final, o vector dos mínimos admissíveis do nível 2 e o n.º do PT em análise.

No contexto desta explicação mais pormenorizada, no módulo da pré-selecção, vai ser apenas considerada a 1ª das quatro tarefas do exemplo. O processo será idêntico para as restantes.

No inicio do módulo é crido o vector de matrizes *DRP*, vector onde vão ser armazenados todos os dados processados neste módulo e que depois será retornado para o módulo principal (Base). Este tipo de vector pode armazenar em cada uma das suas células diferentes tipos de variáveis (*strings*, números absolutos, vectores ou até mesmo matrizes). Este *DRP* terá as dimensões de 6 linhas por *nTare* (n.º de tarefas do PT em análise). Tal como já foi referido, neste módulo a análise vai ser feita tarefa a tarefa e as matrizes criadas vão ser armazenadas, como formato de organização, coluna a coluna. Isto quer dizer que cada coluna do vector de matrizes *DRP* corresponde a uma tarefa, i.e., as matrizes da primeira coluna de *DRP* correspondem à primeira tarefa do plano em análise, e assim sucessivamente.

Após a criação deste vector para armazenamento dos dados é então preenchida a matriz *Drp* com os recursos e respectivos requisitos de nível 1, classificações nos requisitos dos 3 sistemas e CP. É possível verificar na Tabela 2, representada de seguida, os recursos e respectivos dados que foram preenchidos aleatoriamente utilizando as distribuições anteriormente mencionadas. A matriz apresentada está subdividida por cores para uma mais fácil identificação dos dados em questão.

Tabela 2 Matriz *Drp* com dados iniciais dos recursos

Recurso	Requisitos 1º nível				Classificação nos requisitos do sistema da Qualidade				Classificação nos requisitos do sistema Financeiro				Classificação nos requisitos do sistema das Sinergias				CP
1	1	1	1	1	3,0338	3,9595	5,1060	6,2193	8,2575	5,4201	8,5231	6,9508	8,3482	6,2539	4,6864	3,0372	6,3110	7,6371	5,4148	4,9184
2	1	1	1	1	6,0917	5,8724	7,2227	6,2186	9,6280	6,6240	9,6090	4,5538	7,0531	5,4795	7,2003	7,1879	1,6280	3,3459	3,1180	6,8037
3	1	1	1	1	3,8967	6,7949	4,4962	9,0325	5,9349	9,2720	5,1499	7,1789	5,8744	1,9561	4,0357	7,2250	5,8902	3,7625	4,7472	6,4990
4	1	1	1	0	3,8403	6,3984	2,9579	4,5527	4,8135	6,8027	7,8843	6,6010	5,2539	7,6310	7,5978	6,2404	7,1425	6,8256	4,0261	7,5191
5	1	1	1	1	5,7364	7,1907	8,0937	5,6041	6,6554	5,5234	6,4592	6,8800	4,7663	6,5497	7,2022	6,1846	9,4597	4,7829	4,5259	2,5002
6	0	1	1	1	6,3674	6,5816	6,2259	6,8799	6,2033	10,0000	3,6667	2,2914	3,7186	3,8133	5,1328	5,6631	5,5629	7,0827	6,7785	7,5025
7	0	0	1	1	7,8039	2,3287	6,1335	6,0710	10,0000	5,8616	4,9854	6,4716	6,4916	6,1401	4,7828	3,5548	6,6330	3,3143	3,9356	8,6624
8	1	1	1	1	8,0587	5,3099	8,0256	7,2587	5,5740	4,2686	3,9138	5,4599	5,1237	5,1827	7,9671	5,4046	8,2874	4,9368	7,9451	4,9555
9	1	1	1	1	10,0000	7,9020	5,1360	7,2979	5,2798	7,4118	8,8317	2,7910	8,0577	8,9159	6,0949	9,4925	6,3108	3,5258	1,6130	5,3332
10	1	1	1	1	6,2880	2,7227	4,4798	4,3624	7,0395	5,9717	3,6889	5,9810	4,6204	4,6666	7,7283	6,2268	6,7967	7,7679	6,3605	7,1017

O passo seguinte será a avaliação de nível 1, a qual irá eliminar os recursos que não cumpram os requisitos de nível 1 e armazenar os recursos aprovados na matriz *Drp_apr*. Neste exemplo, e observando a Tabela 2 podemos verificar que os recursos 4, 6 e 7 não cumprem esses mesmos requisitos (requisitos não cumpridos assinalados com círculo amarelo). O resultado dessa avaliação é apresentado na tabela seguinte.

Tabela 3 Matriz *Drp_apr* com os recursos aprovados no 1º nível pela ferramenta

Recurso	Classificação nos requisitos do sistema da Qualidade					Classificação nos requisitos do sistema Financeiro					Classificação nos requisitos do sistema das Sinergias					CP
1	3,0338	3,9595	5,1060	6,2193	8,2575	5,4201	8,5231	6,9508	8,3482	6,2539	4,6864	3,0372	6,3110	7,6371	5,4148	4,9184
2	6,0917	5,8724	7,2227	6,2186	9,6280	6,6240	9,6090	4,5538	7,0531	5,4795	7,2003	7,1879	1,6280	3,3459	3,1180	6,8037
3	3,8967	6,7949	4,4962	9,0325	5,9349	9,2720	5,1499	7,1789	5,8744	1,9561	4,0357	7,2250	5,8902	3,7625	4,7472	6,4990
5	5,7364	7,1907	8,0937	5,6041	6,6554	5,5234	6,4592	6,8800	4,7663	6,5497	7,2022	6,1846	9,4597	4,7829	4,5259	2,5002
8	8,0587	5,3099	8,0256	7,2587	5,5740	4,2686	3,9138	5,4599	5,1237	5,1827	7,9671	5,4046	8,2874	4,9368	7,9451	4,9555
9	10,0000	7,9020	5,1360	7,2979	5,2798	7,4118	8,8317	2,7910	8,0577	8,9159	6,0949	9,4925	6,3108	3,5258	1,6130	5,3332
10	6,2880	2,7227	4,4798	4,3624	7,0395	5,9717	3,6889	5,9810	4,6204	4,6666	7,7283	6,2268	6,7967	7,7679	6,3605	7,1017

Nesta fase, as colunas referentes aos requisitos do 1º nível foram eliminadas, devido ao facto de nunca mais serem necessárias para qualquer tipo de análise. Assim, e como foi explicado anteriormente, vai ser criada a matriz *Drp_final*, representada de seguida, a qual vai conter o n.º dos recursos aprovados, as classificações ponderadas de cada um dos sistemas, a classificação ponderada dos 3 sistemas e os CP associados a cada recurso.

Tabela 4 Matriz *Drp_final* com AV dos recursos e classificação ponderada

Recurso	V.S.Q.	V.S.F.	V.S.S.	C.Pond	CP
1	5,3152	7,0992	5,4173	5,9439	4,9184
2	7,0067	6,6639	4,4960	6,0555	6,8037
3	6,0311	5,8862	5,1322	5,6832	6,4990
5	6,6560	6,0357	6,4311	6,3743	2,5002
8	6,8454	4,7897	6,9082	6,1811	4,9555
9	7,1231	7,2016	5,4074	6,5774	5,3332
10	4,9785	4,9857	6,9761	5,6467	7,1017

Para o cálculo das ponderações foram tidos em conta os *inputs* iniciais. Como não foram introduzidos valores na altura do pedido ao utilizador (ver Figura 9) o sistema considera que as ponderações são iguais para os 5 critérios de cada sistema, ou seja, 20% para cada. No caso da ponderação dos 3 sistemas, o caso é idêntico, mas como são considerados apenas 3 sistemas, o valor de ponderação para cada um deles será de 33,3(3)%.

A título de exemplo vamos considerar o recurso 1 e vamos verificar que o valor para o Sistema da Qualidade (V.S.Q.) foi correctamente calculado pelo sistema.

Sendo

r_{ij}; (j = 1, n), (i = 1, k): recurso candidato j para a tarefa i

F.S.Q.: Função objectivo do sistema da qualidade

ΦSQ_i; (i = 1 a 5): ponderação do conjunto de requisitos i do sistema da qualidade

$PQ_i\_r_{ij}$: parâmetro do recurso candidato j à tarefa i para o conjunto de requisitos Qi.

$$(\Phi SQ1 * PQ1\_rij + \Phi SQ2 * PQ2\_rij + \Phi SQ3 * PQ3\_rij + \Phi SQ4 * PQ4\_rij + \Phi SQ5 * PQ5\_rij) = F.S.Q.\_rij \ . \quad (1)$$

Desta forma, e fazendo a substituição pelos valores da Tabela 3 e tendo em consideração que as ponderações são iguais para todos os critérios temos que,

$$(0,2 * 3,0338 + 0,2 * 3,9595 + 0,2 * 5,1060 + 0,2 * 6,2193 + 0,2 * 8,2575) = 5,3152 \ . \quad (2)$$

o que comprova que os cálculos foram de facto bem efectuados pelo algoritmo.

Após este passo, a matriz representada na Tabela 4 vai ser armazenada na 3ª linha (coluna correspondente à tarefa) do vector de matrizes *DRP*, como demonstra a Figura 10, ordenada por ordem decrescente de valor (coluna C.Pond).

```
K>>  DRP

DRP =

                [] [] [] []
                [] [] [] []
    [7x6 double] [] [] []
                [] [] [] []
                [] [] [] []
                [] [] [] []
```

Figura 10 Vector de matrizes *DRP* e matriz armazenada na 3ª linha da 1ª coluna

Desta forma é possível continuar a trabalhar com esta matriz, como forma de optimização do algoritmo por não ser necessário o duplicado de matrizes, sem com isso perder dados importantes para posterior análise.

Agora, já no nível 2 do algoritmo, a matriz vai ser analisada novamente de forma a verificar se todos os recursos aprovados até ao momento têm classificação positiva nos 3 sistemas que estamos a ter em consideração.

Tabela 5 Matriz *Drp_final* aquando do nível 2 do algoritmo de pré-selecção

Recurso	V.S.Q.	V.S.F.	V.S.S.	C.Pond	CP
1	5,3152	7,0992	5,4173	5,9439	4,9184
0	0	0	0	0	0
3	6,0311	5,8862	5,1322	5,6832	6,4990
5	6,6560	6,0357	6,4311	6,3743	2,5002
0	0	0	0	0	0
9	7,1231	7,2016	5,4074	6,5774	5,3332
0	0	0	0	0	0

Tabela 6 Matriz *Drp_final* após análise de 2º nível

Recurso	V.S.Q.	V.S.F.	V.S.S.	C.Pond	CP
1	5,3152	7,0992	5,4173	5,9439	4,9184
3	6,0311	5,8862	5,1322	5,6832	6,4990
5	6,6560	6,0357	6,4311	6,3743	2,5002
9	7,1231	7,2016	5,4074	6,5774	5,3332

Como é possível observar na Tabela 5, em comparação com a matriz representada na Tabela 4, o recurso 2 não possuía classificação positiva no sistema das Sinergias, tal como os recursos 8 e 10 tinham também classificação negativa no sistema Financeiro. Por não preencherem os requisitos de nível 2 (ter classificação positiva nos 3 sistemas) foram eliminados. O próximo passo é ordenar os recursos aprovados por ordem decrescente de valor ponderado do recurso, coluna correspondente à designação de C.Pond das tabelas acima representadas.

Tabela 7 Matriz *Drp_final* ordenada após análise de 2º nível

Recurso	V.S.Q.	V.S.F.	V.S.S.	C.Pond	CP
9	7,1231	7,2016	5,4074	6,5774	5,3332
5	6,6560	6,0357	6,4311	6,3743	2,5002
1	5,3152	7,0992	5,4173	5,9439	4,9184
3	6,0311	5,8862	5,1322	5,6832	6,4990

Após estes passos é feita uma apresentação ao utilizador dos resultados obtidos até ao momento.

```
Para a Tarefa 1

  Referente ao nível 1 do algoritmo (sem avaliação do valor dos recursos)

    Candidataram-se 10 recursos, dos quais foram aprovados 7 no nível 1 do algoritmo de pré-selecção

  Referente ao nível 2 do algoritmo (com avaliação do valor dos recursos)

    6 recursos tiveram nota positiva no S.Q.
    5 recursos tiveram nota positiva no S.F.
    6 recursos tiveram nota positiva no S.S.
    4 recursos tiveram nota positiva nos 3 sistemas

  Recurso    V.S.Q.     V.S.F.     V.S.S.      C.P.
      9      7.1231     7.2016     5.4074     6.5774
      5      6.6560     6.0357     6.4311     6.3743
      1      5.3152     7.0992     5.4173     5.9439
      3      6.0311     5.8862     5.1322     5.6832
```

Figura 11 Exemplo de apresentação de dados de uma tarefa

Após a apresentação dos resultados da tarefa, as matrizes utilizadas para os cálculos vão então ser armazenadas no vector de matrizes *DRP*.

```
DRP =

    [10x21 double]      []      []      []
    [ 7x17 double]      []      []      []
    [ 7x6  double]      []      []      []
                   []   []      []      []
    [ 4x6  double]      []      []      []
                   []   []      []      []
```

Figura 12 Armazenamento das diversas matrizes com os dados dos recursos

Como foi explicitado anteriormente, cada coluna deste *DRP* corresponde aos dados de uma tarefa. Como tal, até este momento apenas a 1ª coluna está a ser preenchida por se tratar da tarefa n.º 1 do PT em questão. Temos armazenada na 1ª linha a matriz *Drp* (Tabela 2) que corresponde aos dados iniciais dos recursos. Na 2ª linha temos a matriz dos recursos aprovados no nível 1 do algoritmo de pré-selecção (Tabela 3), na 3ª linha foi armazenada a matriz *Drp_final* antes de ter sido aplicado o nível 2 do algoritmo de pré-selecção (Tabela 4) e na linha 5 temos a matriz *Drp_final* apenas com os recursos que passaram o 2º nível (Tabela 7).

Após todas as tarefas terem sido analisadas o vector de matrizes *DRP* fica com o aspecto da Figura 13.

DRP =

[10x21 double]	[10x21 double]	[10x21 double]	[10x21 double]
[7x17 double]	[6x17 double]	[7x17 double]	[6x17 double]
[7x6 double]	[6x6 double]	[7x6 double]	[6x6 double]
[]	[]	[]	[]
[4x6 double]	[3x6 double]	[4x6 double]	[5x6 double]
[]	[]	[]	[]

Figura 13 Vector de matrizes *DRP* após análise de todas as tarefas

Por ser um requisito do módulo de selecção final o n.º de recursos aprovados em cada nível tem de ser igual para todas as tarefas dentro do mesmo PT, o que significa que as matrizes terão de ser modificadas para que fiquem todas com a mesma dimensão. Na 3ª linha do vector *DRP* podemos verificar pela imagem acima apresentada que o n.º mínimo de recursos que passam o 1º nível do algoritmo de pré-selecção, *k_min1*, é 6 (seis). As matrizes da 3ª linha serão copiadas para a 4ª linha do vector *DRP* e, a partir da dimensão mínima (6 recursos), as linhas das matrizes serão preenchidas com 0 (zeros). De seguida as linhas preenchidas com zeros serão eliminadas ficando todas as matrizes com as mesmas dimensões.

O mesmo será feito para as matrizes referentes ao 2º nível do algoritmo. Desta forma as matrizes da 5ª linha do vector *DRP* serão copiadas para a 6ª linha. A partir da dimensão mínima três recursos, *k_min2*, as restantes linhas serão preenchidas com zeros e eliminadas (ver Tabela 8 e Tabela 9 referentes ao redimensionamento da matriz da tarefa 1 para o 2º nível do algoritmo).

A matriz que vai sofrer o redimensionamento é a correspondente à representada na Tabela 7. Como o n.º mínimo é de três recursos, *k_min2*, essa mesma matriz, que foi copiada para a 6ª linha do vector *DRP* ficará com o seguinte aspecto

Tabela 8 Matriz 6ª linha de *DRP* no processo de redimensionamento para o *k_min2*

Recurso	V.S.Q.	V.S.F.	V.S.S.	C.Pond	CP
9	7,1231	7,2016	5,4074	6,5774	5,3332
5	6,6560	6,0357	6,4311	6,3743	2,5002
1	5,3152	7,0992	5,4173	5,9439	4,9184
0	0	0	0	0	0

Tabela 9 Matriz 6ª linha do vector *DRP* redimensionada para o *k_min2*

Recurso	V.S.Q.	V.S.F.	V.S.S.	C.Pond	CP
9	7,1231	7,2016	5,4074	6,5774	5,3332
5	6,6560	6,0357	6,4311	6,3743	2,5002
1	5,3152	7,0992	5,4173	5,9439	4,9184

No final deste processo para todas as tarefas, o aspecto final do vector de matrizes *DRP* será o seguinte:

```
DRP =

[10x21 double]    [10x21 double]    [10x21 double]    [10x21 double]
[ 7x17 double]    [ 6x17 double]    [ 7x17 double]    [ 6x17 double]
[ 7x6  double]    [ 6x6  double]    [ 7x6  double]    [ 6x6  double]
[ 6x6  double]    [ 6x6  double]    [ 6x6  double]    [ 6x6  double]
[ 4x6  double]    [ 3x6  double]    [ 4x6  double]    [ 5x6  double]
[ 3x6  double]    [ 3x6  double]    [ 3x6  double]    [ 3x6  double]
```

Figura 14 Vector de matrizes *DRP* totalmente preenchido

Este será um dos outputs do módulo da pré-selecção. O outro *output* é o vector *CP* dos recursos que passam o 1º nível do algoritmo de pré-selecção. A criação deste vector é bastante simples. Basta aceder a cada uma das matrizes da 4ª linha do vector de matrizes *DRP* e copiar a última coluna, correspondente ao CP do recurso. Tal é feito utilizando as poucas linhas de código apresentadas de seguida:

```
CP=[];
for jj=1:nTare
    Cp=DRP{4,jj}(:,6)';
    CP=[CP, Cp];
end
```

Na figura seguinte estão representadas duas iterações que apresentam as matrizes dos recursos que foram aprovados no 1º nível do algoritmo de pré-selecção, 1ª e 2ª tarefas respectivamente, e o vector dos *CP* já completo (embora não seja possível ver todos os seus elementos). Como é facilmente identificável, e sabendo que a última coluna de cada matriz corresponde ao CP associado ao recurso, é possível verificar que o vector *CP* está a ser correctamente preenchido.

```
>>  DRP{4,1}

ans =

    9.0000    7.1231    7.2016    5.4074    6.5774    5.3332
    5.0000    6.6560    6.0357    6.4311    6.3743    2.5002
    8.0000    6.8454    4.7897    6.9082    6.1811    4.9555
    2.0000    7.0067    6.6639    4.4960    6.0555    6.8037
    1.0000    5.3152    7.0992    5.4173    5.9439    4.9184
    3.0000    6.0311    5.8862    5.1322    5.6832    6.4990

>>  DRP{4,2}

ans =

    4.0000    4.8691    6.4578    7.5724    6.2998    8.1269
    9.0000    6.9982    6.3518    5.3414    6.2305    2.8990
   10.0000    5.4388    6.6651    6.0950    6.0663    6.8022
    5.0000    5.1126    5.6815    5.9534    5.5825    3.2295
    6.0000    7.0753    4.9980    4.6014    5.5583    7.5591
    3.0000    3.9220    6.2440    6.2969    5.4877    2.7454

>> CP

CP =

  Columns 1 through 10

    5.3332    2.5002    4.9555    6.8037    4.9184    6.4990    8.1269    2.8990    6.8022    3.2295

  Columns 11 through 20

    7.5591    2.7454    4.3805    3.8938    6.8184    5.5909    5.3534    7.0817    5.1015    4.7628
```

Figura 15 Preenchimento do vector dos CP

Estando o vector dos CP preenchido é então retornado, em conjunto com o vector de matrizes *DRP*, para o módulo principal (Base).

Selecção final

De volta ao módulo principal vai ser verificado o n.º de recursos que passam o 1º nível da pré-selecção (*k_min1*) que depois, aquando da chamada do módulo de selecção final, será parâmetro de entrada, juntamente com o vector *CP* e o n.º de tarefas (*nTare*).

```
    k_min1=size(DRP{4,1},1);

    [x1, fx1, exitflag1, output1,
CT]=Selec_BB(nTare(ii),k_min1,CP);
```

Como parâmetros de saída do módulo de selecção final, tendo em conta que esta será a selecção sem ter em conta a AV dos recursos, teremos o vector *x1*, a partir do qual conseguiremos identificar os recursos seleccionados e os CT associados; a variável *fx1* onde virá guardado o custo total do sistema; a variável *exitflag1* cujo valor nos permite saber se a solução encontrada é óptima ou não; e uma estrutura, *output1*, com diversos

campos referentes ao n.º de iterações efectuadas nos cálculos, o tempo utilizado na simulação, entre outras.

Como já foi mencionado anteriormente, este módulo não será explicado tão pormenorizadamente como o resto do programa uma vez que já estava desenvolvido. Será mais relevante mencionar que este módulo utiliza o algoritmo do tipo *Branch&Bound* como algoritmo de optimização.

É dentro do módulo de selecção final que o vector *CT* é criado (com as dimensões de (*nTare*-1)**k_min1*k_min1*). Após a criação do vector dos CT, vai ser criado um vector designado por *f* que vai ser a concatenação/junção do vector *CP* com o vector *CT*. Este vector vai ser um parâmetro de entrada para a função que vai realmente seleccionar os recursos. A escolha dos recursos vai ser optimizada tendo em conta os CP e os CT.

Tal como já foi referido, o módulo da selecção retorna o vector *x1*. Este vector tem a particularidade de ter as dimensões do vector *f* referido anteriormente e ser do tipo boleano. Isto significa que este vector é apenas preenchido com 0 (zeros) e 1 (um), em que o 1 significa selecção.

Tendo em conta o caso concreto que tem vindo a ser analisado, em que o *k_min1* é de seis recursos, vamos ter um *CP* com a dimensão de 24 (4 tarefas por 6 recursos). Isto significa que cada 1 encontrado nas 24 primeiras posições do vector *x1* corresponde a um CP seleccionado para uma tarefa, que por sua vez está associado a um recurso. Os 1 encontrados nas restantes posições dizem respeito ao CT entre os recursos seleccionados para as tarefas. Assim, o primeiro 1 encontrado, fora das 24 primeiras posições, diz respeito ao CT entre a tarefa 1 e a tarefa 2, o segundo diz respeito ao CT entre a tarefa 2 e a tarefa 3 e assim sucessivamente.

Para que de uma forma mais expedita se encontrem as posições dos recursos seleccionados usa-se a função *find* [11] do MATLAB para encontrar as posições do vector *x1* que contêm 1 (um) e armazenam-se os resultados no vector *locali*. De notar que as posições armazenadas são absolutas e é necessário encontrar as posições dos recursos relativas a cada tarefa. No exemplo que tem vindo a ser explicado, usando a expressão seguinte obtém-se o vector apresentado na Figura 16.

```
Locali=find(x1==1)
```

Figura 16 Vector *locali* com posições dos recursos seleccionados

No caso dos elementos salientados a cor vermelha, correspondentes às posições dos CP escolhidos, associados a um determinado recurso, verificamos que na primeira posição se encontra o valor 2. Neste caso, e como se trata da primeira tarefa, o valor 2 tem correspondência directa à posição do recurso na matriz dos recursos referente à primeira tarefa (coluna 1, linha 4 do vector de matrizes *DRP*), correspondendo neste caso ao recurso 5 (conforme assinalado na 1ª coluna da Figura 17). No caso do 2º valor, 8, já é necessário proceder a alguns cálculos, nomeadamente retirar ao valor absoluto um múltiplo de *k_min1* (de valor 6 para este caso concreto) para que se fique com a posição do recurso apenas referente à 2ª tarefa. Neste caso o recurso seleccionado será novamente o que está na 2ª posição da matriz referente à 2ª tarefa (coluna 2, linha 4 do vector de matrizes *DRP*), sendo o recurso 9 (nove) o seleccionado (2ª matriz representada na Figura 17).

Para o caso do 3º valor, 14, é necessário tirar o n.º de recursos das duas tarefas anteriores (2*6), o que nos dará novamente o 2º recurso na matriz da 3ª tarefa. No caso do 4º valor é necessário retirar o n.º de recursos das 3 tarefas anteriores (18) dando como resultado a 6ª posição da matriz referente à 4ª tarefa.

Com as posições dos recursos encontradas, o n.º do recurso, a classificação ponderada e o CP associado a esse recurso vai ser armazenado na 1ª célula de outro vector de matrizes de nome *valor*[1].

[1] O vector de matrizes *valor* irá guardar os dados referentes a mais do que um PT, caso seja definido no início. Na Figura 18 está representado o vector apenas para o exemplo em questão. Caso a simulação abranja mais do que 1 PT, os resultados serão armazenados da mesma forma mas em colunas diferentes (resultados de cada PT são guardados nas respectivas colunas)

```
>> DRP{4,1}

ans =

    9.0000    7.1231    7.2016    5.4074    6.5774    5.3332
    5.0000    6.6560    6.0357    6.4311    6.3743    2.5002
    8.0000    6.8454    4.7897    6.9082    6.1811    4.9555
    2.0000    7.0067    6.6639    4.4960    6.0555    6.8037
    1.0000    5.3152    7.0992    5.4173    5.9439    4.9184
    3.0000    6.0311    5.8862    5.1322    5.6832    6.4990

>> DRP{4,2}

ans =

    4.0000    4.8691    6.4578    7.5724    6.2998    8.1269
    9.0000    6.9982    6.3518    5.3414    6.2305    2.8990
   10.0000    5.4388    6.6651    6.0950    6.0663    6.8022
    5.0000    5.1126    5.6815    5.9534    5.5825    3.2295
    6.0000    7.0753    4.9980    4.6014    5.5583    7.5591
    3.0000    3.9220    6.2440    6.2969    5.4877    2.7454

>> DRP{4,3}

ans =

    7.0000    7.3641    6.5484    6.1106    6.6743    4.3805
   10.0000    6.2799    7.7712    4.5040    6.1851    3.8938
    5.0000    6.4435    5.9543    5.9823    6.1267    6.8184
    6.0000    6.2969    5.7011    6.0554    6.0178    5.5909
    8.0000    6.6797    5.6699    5.2741    5.8746    5.3534
    2.0000    4.6783    5.6014    6.6621    5.6473    7.0817

>> DRP{4,4}

ans =

    8.0000    6.2636    6.1886    6.7580    6.4034    5.1015
    9.0000    5.9717    6.4221    6.0769    6.1569    4.7628
    5.0000    6.2932    5.9528    5.6323    5.9594    7.1062
   10.0000    6.2918    6.2620    5.2899    5.9479    4.6905
    1.0000    6.2242    5.2225    5.6303    5.6923    6.5391
    6.0000    4.8990    5.0826    4.9355    4.9724    6.0347
```

Figura 17 Recursos seleccionados sem ter em conta a análise de valor

Todo esse processo é realizado pelo código abaixo apresentado.

```
for jj=1:nTare(ii)
        posi=locali(jj)-k_min1*(jj-1);
%localização do recurso para cada tarefa
        valor{1,ii}(jj,1)=DRP{4,jj}(posi,1);
%copia o n.º do recurso
        valor{1,ii}(jj,2)=DRP{4,jj}(posi,5);
%copia a classificação final do recurso
        valor{1,ii}(jj,3)=DRP{4,jj}(posi,6);
%copia os custos de processamento do recurso
end
```

A cada iteração a posição do recurso é identificada e os dados são copiados para a 1ª célula do vector de matrizes *valor*, conforme Figura 18.

```
>> valor

valor =

    [4x3   double]
    [     23.7622]
    [1x24  double]
    [1x108 double]
    [     27.2808]

>> valor{1}

ans =

    5.0000    6.3743    2.5002
    9.0000    6.2305    2.8990
   10.0000    6.1851    3.8938
    6.0000    4.9724    6.0347
```

Figura 18 Vector de matrizes *valor* e 1ª célula (matriz dos recursos seleccionados)

A segunda célula corresponde ao valor do sistema, que consiste na soma do valor dos recursos seleccionados identificados pelo rectângulo verde. A terceira e quarta células correspondem ao vector dos CP e dos CT respectivamente, enquanto que a 5ª célula corresponde ao custo total do sistema seleccionado (CP+CT), *fx1*.

Apresentação de resultados

A primeira parte da apresentação serve apenas para relembrar ao utilizador alguns aspectos relativos ao PT em questão e relembrar o n.º de recursos que passou o 1º nível do algoritmo de pré-selecção.

De seguida é apresentado o sistema de recursos escolhido. Como o utilizador não observa nenhum dos passos descritos depois da introdução dos dados e não sabe como estes estão organizados é necessário representá-los de uma forma que seja de fácil interpretação.

```
for kk=1:size(valor{1,pp},1)
  varname=['R' num2str(kk) ','
num2str(valor{1,pp}(kk,1)) '; '];
   fprintf(varname);
end
```

O código representado neste excerto é responsável pela apresentação do sistema de recursos escolhido. A variável temporária *varname* é uma concatenação/junção da letra R

com o número da tarefa (que vai variar com a variável *kk* do ciclo *for*) e com o n.º do recurso seleccionado para a tarefa em questão (*kk*). Assim, e pela imagem seguinte temos que para a 1ª tarefa foi seleccionado o recurso 5, para a 2ª tarefa foi seleccionado o recurso 9 e assim sucessivamente.

De seguida, para a apresentação dos custos do sistema é apenas necessário aceder à 5ª célula do vector de matrizes *valor*, na qual foram armazenados os custos totais do sistema seleccionado. Para a descrição desses custos do sistema, vão ser apresentadas as parcelas de CP e de CT. Para a obtenção do valor total dos CP é apenas necessário fazer a soma dos valores realçados a azul na Figura 18. Essa soma e apresentação dos CP é possível devido à seguinte linha de código:

```
fprintf('\n\t\tCP:
%.4f\n',sum(valor{1,pp}(:,3)));
```

A expressão `sum(valor{1,pp}(:,3))` é a responsável pela soma dos valores daquela coluna mencionada.

São também apresentadas todas as parcelas desse valor total de forma a indicar ao utilizador os custos associados a cada recurso seleccionado. Isto é feito de forma semelhante à apresentação do sistema de recursos.

Os CT totais são obtidos por forma de uma simples operação matemática. Aos custos totais armazenados na 5ª célula do vector *valor* vão ser subtraídos os CP totais calculados anteriormente. Para apresentação individual de cada CT é necessário aceder ao vector *locali*, representado na Figura 16. Começando a percorrer as posições assinaladas a azul, é necessário, a cada uma delas retirar o valor da dimensão do vector *CP*. Ao fazermos isso, os valores com que ficamos dizem respeito aos CT de umas tarefas para as outras. O 1º valor diz respeito ao CT da tarefa 1 para a tarefa 2, o 2º valor diz respeito ao CT da tarefa 2 para a tarefa 3 e assim sucessivamente, qualquer que seja o n.º de tarefas.

A apresentação do valor do sistema é obtido pelo mesmo processo descrito para os CP, tal como a forma de apresentação dos valores individuais, variando apenas a coluna a aceder da matriz armazenada no vector *valor* (coluna verde).

Como esta selecção ainda não contempla a AV dos recursos é verificado o n.º de recursos que tem classificação negativa (abaixo de 5). No nosso exemplo podemos verificar que foi escolhido um recurso com classificação negativa, como é possível constatar na Figura 19.

É de seguida apresentado, em forma de texto, o resultado retornado pela *exitflag*, valor de controlo para que o utilizador possa saber se a solução apresentada é óptima, ou se pelo contrário ocorreu algum tipo de contratempo, como o tempo de simulação ou o n.º de iterações máximas definidas terem sido excedidos.

São apresentadas os n.º de iterações realizadas para encontrar a solução apresentada, bem como o tempo, de forma amigável para o utilizador, utilizado na simulação (valores armazenados na estrutura de dados *output1*).

```
------------------------------------------------------------------
Resultados da selecção final para o Plano de Tarefas 1

N.º de tarefas: 4
N.º inicial de recursos candidatos: 10
k minimo do nível 1: 6

Sem análise de valor

    Sistema de recursos: R1,5; R2,9; R3,10; R4,6;

    Custo do sistema: 27.2808
        CP: 15.3277
            CP R1,5= 2.5002;
            CP R2,9= 2.899;
            CP R3,10= 3.8938;
            CP R4,6= 6.0347;

        CT: 11.9531
            CT 1->2= 4.1147
            CT 2->3= 3.8495
            CT 3->4= 3.9889

    Valor do sistema: 23.7622
        Valor R1,5= 6.3743;
        Valor R2,9= 6.2305;
        Valor R3,10= 6.1851;
        Valor R4,6= 4.9724;

Nº de recursos seleccionados com classificação negativa: 1

Foi encontrada a solução óptima

N.º de iterações efectuadas: 807

Tempo da simulação = 0 horas,0 minutos e 13 segundos
```

Figura 19 Apresentação de resultados sem ter em conta a AV

De seguida, após a apresentação de resultados ter sido efectuada, voltamos ao módulo principal. O n.º de recursos que passam o 2º nível da pré-selecção (6ª linha do vector de matrizes *DRP*) vai ser verificado. Se nenhum recurso tiver tido aprovação o programa termina aqui (caso não existam mais PT a analisar), caso contrário é necessário fazer o redimensionamento dos vectores de CP e CT. No entanto é necessário ter atenção à forma como este redimensionamento é efectuado. No que diz respeito aos CP não há qualquer tipo de complicação uma vez que estes "acompanham" sempre os recursos a que estão associados. Isto significa que basta aceder à última coluna da matriz dos recursos aprovados no 2º nível de cada tarefa e proceder de forma idêntica ao que foi explicado para a criação do primeiro vector dos CP guardando o novo vector dos CP com o nome de *CP2* para que não se percam os dados que podem ser úteis para alguma análise.

No que diz respeito aos CT o caso é mais complicado. Neste vector temos as várias combinações possíveis de transporte entre as tarefas. Tal como está exemplificado na Figura 20 (apenas entre duas tarefas) temos 36 combinações possíveis de transporte (tendo em conta que passam 6 recursos no 1º nível, *k_min1*, tal como no exemplo que está a ser explicado)

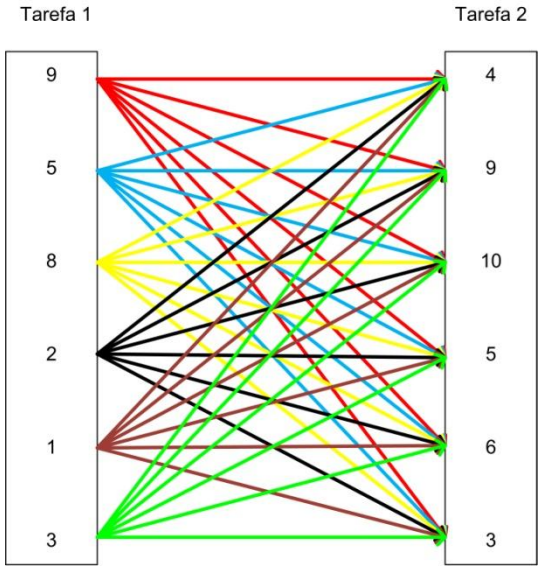

Figura 20 Ligações dos CT entre as tarefas 1 e 2[2]

[2] De notar que os números apresentados em cada uma das tarefas correspondem aos números dos recursos que foram aprovados no nível 1 do algoritmo de pré-selecção e que foram tidos em conta para a criação do vector inicial dos *CT*

Ao ser efectuado o 2º nível do algoritmo de pré-selecção (com AV) alguns dos recursos serão eliminados, e consequentemente as ligações referentes a esses recursos deixarão também de existir. Tendo em conta também, e como já foi mencionado, o *k_min2*, passam apenas 3 recursos no final do 2º nível. Como tal, o n.º de ligações entre recursos das diferentes tarefas reduz significativamente. No exemplo em questão, entre a tarefa 1 e a tarefa 2, deixa-se de ter 36 ligações para se passar a ter apenas 9 como é visível na imagem seguinte. O mesmo se passará entre as restantes tarefas (tarefa 2 com tarefa 3 e tarefa 3 com tarefa 4)

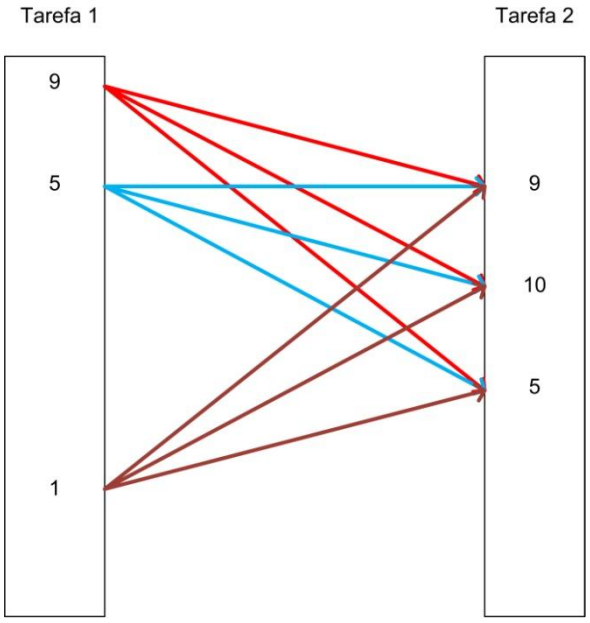

Figura 21 Ligações dos CT entre tarefas após análise de valor

Na criação do novo vector dos CT é necessário ter especial atenção para que se mantenha a coerência entre os valores a ser guardados e as ligações ainda existentes.

Para que tal seja possível é necessário, em primeiro lugar, proceder a uma reestruturação do vector *CT* dos valores correspondentes às duas primeiras tarefas. Assim, e sabendo que os primeiros 36 valores correspondem aos diversos custos de transporte entre a tarefa 1 e 2, vão ser ordenados em forma de uma matriz de 6 por 6 (ou *k_min1* por *k_min1*) como a que está representada de seguida.

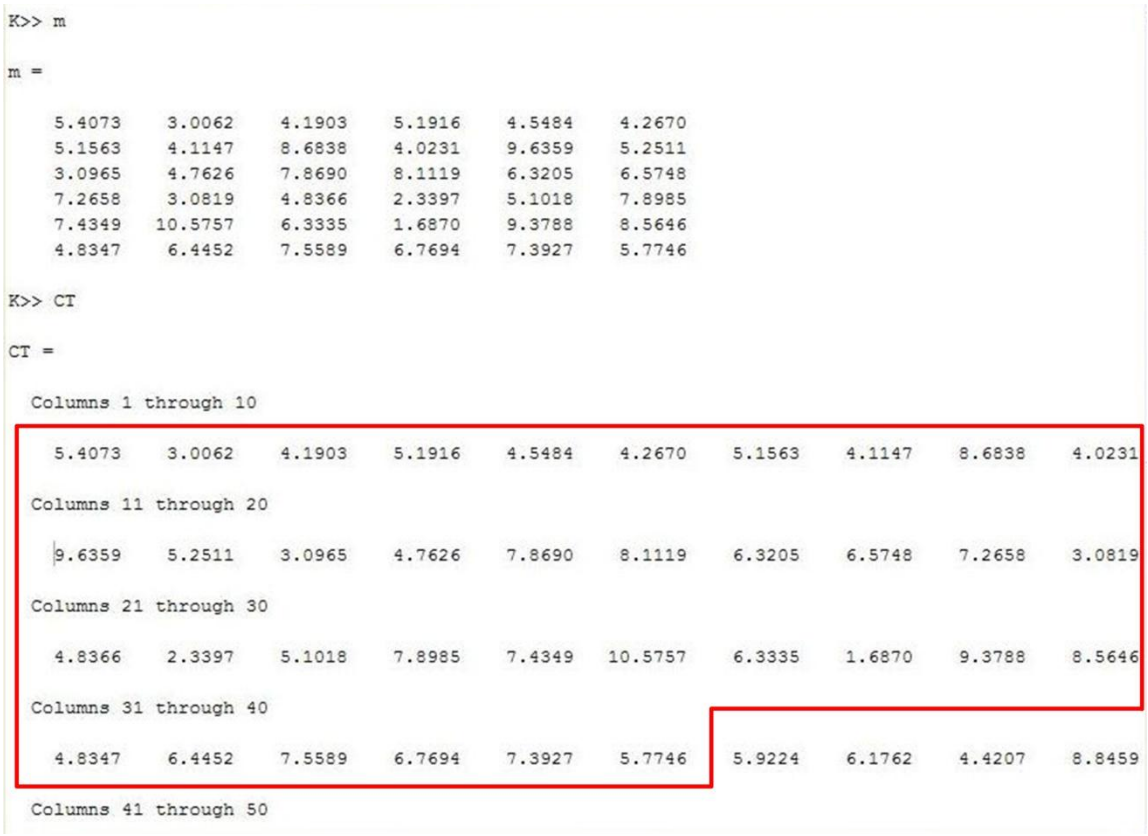

Figura 22 Matriz com os valores de CT entre a tarefa 1 e 2 reestruturados

Estando este passo efectuado é necessário identificar os recursos que passam do 1º para o 2º nível do algoritmo, referente à mesma tarefa. Para isso, são criados 2 vectores, *a* e *b*, que vão armazenar os recursos que passam o 1º e o 2º nível respectivamente. De seguida será criado novo vector, *pos*, com as dimensões do vector *a*. Este vector será, no inicio, preenchido com zeros. Quando for feita a comparação entre os vectores dos recursos aprovados em cada um dos níveis, será colocado o valor 1 (um) quando o n.º de um recurso for identificado nos 2 vectores (*a* e *b*). No vector *pos*, o valor 1 ficará na posição relativa em que o recurso comum aos 2 vectores está no vector *a*. No final da comparação será criado novo vector, de nome *pos2* que irá conter as posições do vector *pos* que contém o valor 0 (zero), o que significa que os recursos que estavam nessa posição no final do 1º nível do algoritmo de pré-selecção foram eliminados no 2º nível. O que foi descrito aqui está representado na figura seguinte.

```
K>> a

a =

     9     5     8     2     1     3

K>> b

b =

     9     5     1

K>> pos

pos =

     1     1     0     0     1     0

K>> pos2

pos2 =

     3     4     6
```

Figura 23 Vectores de verificação de eliminação de recursos

Com estes dados podemos então eliminar os CT referentes aos recursos da 1ª tarefa que foram eliminados. Assim sendo, as linhas[3] indicadas no vector *pos2* vão ser preenchidas com 0 (zero), Figura 24.

[3] Cada linha da matriz *m*, neste exemplo, corresponde a um recurso. Assim sendo, e tendo em conta o vector *a* apresentado na Figura 23, correspondente aos recursos que passam o 1º nível da pré-selecção, a primeira linha da matriz *m* corresponde aos custos relacionados com o recurso 9, a segunda linha com o recurso 5 e assim sucessivamente.

```
K>> a

a =

     9     5     8     2     1     3

K>> b

b =

     9     5     1

K>> pos2

pos2 =

     3     4     6

K>> m

m =

    5.4073    3.0062    4.1903    5.1916    4.5484    4.2670
    5.1563    4.1147    8.6838    4.0231    9.6359    5.2511
         0         0         0         0         0         0
         0         0         0         0         0         0
    7.4349   10.5757    6.3335    1.6870    9.3788    8.5646
         0         0         0         0         0         0
```

Figura 24 Eliminação dos CT referentes aos recursos eliminados da 1ª tarefa

Vão ser agora eliminados os CT referentes aos recursos que foram eliminados na passagem da análise do 1º para o 2º nível da tarefa 2.

O processo será idêntico ao explicado para a eliminação dos CT da tarefa 1 com a particularidade de neste caso, cada um dos recursos da tarefa 2 ser organizado por colunas. Sendo assim, e de acordo com o vector *a* da Figura 25, a 1ª coluna corresponde ao recurso 4, a 2ª coluna corresponde ao recurso 9 e assim sucessivamente.

Como se pode verificar, com este tipo de processo, conseguem-se eliminar os valores de CT que estavam em excesso de uma forma relativamente simples e eficaz. Tal como era expectável, é possível verificar pela Figura 25 que restaram 9 valores para os CT.

O próximo passo é copiar estes valores para o novo vector dos CT que irá ser denominado, à semelhança do que aconteceu ao vector dos CP, *CT2*.

```
K>> a

a =

     4     9    10     5     6     3

K>> b

b =

     9    10     5

K>> pos

pos =

     0     1     1     1     0     0

K>> pos2

pos2 =

     1     5     6

K>> m

m =

         0    3.0062    4.1903    5.1916         0         0
         0    4.1147    8.6838    4.0231         0         0
         0         0         0         0         0         0
         0         0         0         0         0         0
         0   10.5757    6.3335    1.6870         0         0
         0         0         0         0         0         0
```

Figura 25 Eliminação dos CT referentes aos recursos eliminados da 2ª tarefa

Para preencher o novo vector *CT2* vai ser usada a função *find* do MATLAB.

```
x=find(m~=0)';
Ct2=zeros(1,size(x,2));
    for kk=1:size(x,2)
        Ct2(kk)=m(x(kk));
    end
CT2=[CT2, Ct2];
```

Desta forma é mais fácil encontrar as posições em que estão os valores que têm de ser guardados. No entanto existe um pequeno detalhe com a função *find*, quando se trata de matrizes. A pesquisa por valores, neste caso, diferentes de 0 (zero) é feita coluna a coluna e não linha a linha como seria de esperar. Sendo assim, e para contornar este problema, antes de o programa efectuar as linhas de código acima apresentadas é necessário modificar a matriz *m* para a sua transposta, ou seja, as linhas passam a ser colunas e vice-versa, Figura 26.

```
K>> m

m =

         0    3.0062    4.1903    5.1916         0         0
         0    4.1147    8.6838    4.0231         0         0
         0         0         0         0         0         0
         0         0         0         0         0         0
         0   10.5757    6.3335    1.6870         0         0
         0         0         0         0         0         0

K>> m'

ans =

         0         0         0         0         0         0
    3.0062    4.1147         0         0   10.5757         0
    4.1903    8.6838         0         0    6.3335         0
    5.1916    4.0231         0         0    1.6870         0
         0         0         0         0         0         0
         0         0         0         0         0         0
```

Figura 26 Matriz *m* e matriz *m* transposta

O que se pretende é passar os valores linha a linha, pela mesma ordem em que estão representados na 1ª matriz da Figura 26, pois é considerado que o vector dos *CT* está organizado desta forma. Neste caso, em que temos 3 ligações possíveis para cada recurso, estabeleceu-se que os 3 primeiros valores (1ª linha da matriz *m*) correspondem a todas a ligações possíveis entre o 1º recurso da tarefa 1 (recurso 9) e os recursos da tarefa 2 (recurso 9, 10 e 5). Estes valores correspondem às ligações representadas a cor vermelha na Figura 21. Na 2ª linha da matriz *m*, correspondente ao 2º recurso da tarefa 1 (recurso 5), temos, da mesma forma, as ligações aos recursos 9, 10 e 5 da tarefa 2, ligações representadas a cor azul na Figura 21. O mesmo acontece para a 3ª linha. Assim, para que tudo fique coerente, e pelo modo de funcionamento da função *find* os valores vão ser copiados coluna a coluna, motivo que origina a criação da matriz *m* transposta, Figura 27.

```
K>> CT2

CT2 =

     []

K>> m

m =

         0         0         0         0         0         0
    3.0062    4.1147         0         0   10.5757         0
    4.1903    8.6838         0         0    6.3335         0
    5.1916    4.0231         0         0    1.6870         0
         0         0         0         0         0         0
         0         0         0         0         0         0

K>> CT2

CT2 =

    3.0062    4.1903    5.1916    4.1147    8.6838    4.0231   10.5757    6.3335    1.6870
```

Figura 27 Matriz *m* transposta e 1ª parte do preenchimento do vector *CT2*

Para o preenchimento do resto do vector o procedimento é idêntico. Desta forma, os novos vectores *CT2* e *CP2* ficam com as dimensões de 27 e 12 respectivamente. É agora possível proceder à selecção dos recursos tendo em conta a AV.

Os passos para esta selecção são em tudo idênticos aos explicados para a selecção final sem AV. Apenas mudam os nomes de algumas variáveis que são retornadas pelo módulo de selecção final para que não haja conflito com as retornadas aquando da primeira vez que o módulo foi chamado pelo programa, bem como o vector de matrizes que armazena os resultados, agora de nome *valor2*.

Após a apresentação dos resultados da selecção final com a AV é apresentado o tempo total da simulação (soma do tempo das simulações da selecção final com e sem AV) como é possível observar na figura seguinte.

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
N.º inicial de recursos candidatos: 10
k minimo do nível 1: 6
k minimo do nível 2: 3

Com análise de valor

    Sistema de recursos: R1,5; R2,9; R3,7; R4,5;

    Custo do sistema: 30.3663
        CP: 16.8859
            CP R1,5= 2.5002;
            CP R2,9= 2.899;
            CP R3,7= 4.3805;
            CP R4,5= 7.1062;

        CT: 13.4804
            CT 1->2= 4.1903
            CT 2->3= 4.2901
            CT 3->4= 4.9999

    Valor do sistema: 25.2385
        Valor R1,5= 6.3743;
        Valor R2,9= 6.2305;
        Valor R3,7= 6.6743;
        Valor R4,5= 5.9594;

Foi encontrada a solução óptima

N.º de iterações efectuadas: 183

Tempo da simulação = 0 horas,0 minutos e 1 segundos
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Tempo total da simulação = 0 horas,0 minutos e 14 segundos
```

Figura 28 Apresentação dos resultados tendo em conta a AV

O nome do *DRP*, vector de matrizes que armazena os dados dos recursos referentes ao PT analisado, é alterado para que, no caso de nos *inputs* se ter considerado a análise de mais do que um PT, os dados não se percam, ou seja, para *nPT*=2 vamos obter dois vectores de matrizes, *DRP1* e *DRP2*, para o 1º PT e para o 2º PT respectivamente.

No Anexo A é representado um exemplo da apresentação de resultados com todos os outputs gerados para outra simulação efectuada com parâmetros diferentes do exemplo explicado neste capítulo. Nesta simulação foram considerados 2 PT, para comprovar a flexibilidade do programa construído, permitindo efectuar análises a mais do que um PT.

Esta ferramenta/demonstrador já efectuou um conjunto de simulações como forma de suporte/confirmação das vantagens de um modelo que contempla uma pré-selecção com integração de AV. Pretendia-se validar o pressuposto que a integração da AV na fase de pré-selecção incorpora mais-valias ao processo global da selecção de recursos, nomeadamente, através do alcançar de um maior valor do sistema final seleccionado. Através destas simulações já se puderam verificar alguns pressupostos do modelo de pré-selecção objecto de simulação, tais como o maior valor do sistema seleccionado com integração AV e a garantia de que todos os recursos seleccionados têm classificação positiva, tendo por comparação os resultados obtidos nas simulações sem ter em conta a AV. Outra conclusão relevante que estas simulações com o nosso demonstrador permitiu aferir diz respeito ao menor período de tempo utilizado pelo algoritmo para essa selecção com integração AV.

6. CONCLUSÕES E DESENVOLVIMENTOS FUTUROS

Este capítulo, referente às conclusões, vai ser dividido em duas partes. A primeira é referente às conclusões gerais no âmbito desta dissertação e a segunda diz respeito ao trabalho a ser realizado futuramente no que concerne ao desenvolvimento da ferramenta.

6.1. CONCLUSÕES GERAIS

Com base no BM_VEARM, o modelo de selecção contempla duas fases principais, a pré-selecção e selecção de recursos. No âmbito desta dissertação teve-se como principal objectivo o desenvolvimento de uma ferramenta de pré-selecção de recursos. Para que tal fosse possível, foi necessário compreender a pré-selecção de recursos para empresas distribuídas, desenvolver o algoritmo de pré-selecção em pseudo-código, implementar o algoritmo em código MATLAB, integrar com um módulo de selecção final e finalmente simular e validar o código criado.

Apesar de existirem diversas ferramentas que poderiam ter sido utilizadas para a elaboração do algoritmo de pré-selecção, a escolha recaiu sobre o MATLAB devido às

características mencionadas no Capítulo 4 e pelo facto do algoritmo de selecção final de recursos já estar implementado em MATLAB. Continuando a trabalhar com a mesma plataforma de desenvolvimento torna-se significativamente mais fácil a integração do algoritmo da pré-selecção com a selecção final.

Para cumprir com os objectivos propostos foram ultrapassados alguns obstáculos, nomeadamente a forma como o MATLAB organiza e armazena os dados (vectores e matrizes) e a habituação à linguagem MATLAB, embora seja parecida com a linguagem C. No entanto, o principal obstáculo encontrado foi a integração com a selecção final que obrigou a uma profunda reflexão para ser encontrada uma forma de eliminar apenas as ligações dos CT relacionados com os recursos que foram eliminados do 1º para o 2º nível da pré-selecção nas diversas tarefas.

Esta ferramenta foi validada com cálculos que se demonstraram nesta dissertação e teve já uma aplicação real, tendo já realizado um determinado plano de simulações com vista a validar um modelo de pré-selecção com integração AV em desenvolvimento numa tese de doutoramento.

Como conclusão final, constatou-se que os objectivos propostos foram atingidos na íntegra, tendo sido uma fonte de motivação pessoal o desenvolvimento do trabalho desta dissertação.

6.2. DESENVOLVIMENTOS FUTUROS

O código produzido nesta ferramenta, devido à pouca familiaridade inicial com o programa MATLAB, pode sofrer algumas alterações como forma de melhoria:

- Optimização do código pois, à medida que o programa ia sendo desenvolvido e os conhecimentos aumentavam algumas funcionalidades já foram melhoradas. No entanto, no que diz respeito às fases iniciais, o código ainda poderá ser mais optimizado;

- Criação de uma interface gráfica mais amigável para o utilizador para introdução dos dados iniciais;

- Possibilitar a escolha de mais do que um algoritmo de selecção final preparando o algoritmo de pré-selecção para ser integrado em qualquer tipo de algoritmo de selecção final;

- Importação e exportação de dados de folhas Excel para que depois possam ser criados gráficos para análise de resultados, tudo isto de forma autónoma.

Referências Documentais

[1] Ávila, P., Costa, L., Bastos, J., Lopes, P., Pires, A. – *Analysis of the Domain of Applicability of an Algorithm for a Resources System Selection Problem for Distributed/Agile/Virtual Enterprises Integration*, Sistemas y Tecnologías de Información, Actas de la 5ª Conferencia Ibérica de Sistemas y Tecnologías de Información, Santiago de Compostela, España, Junho 2010, Vol.I, pag.506-510

[2] Ávila, P. – *Modelo Rigoroso de Selecção de Sistemas de Recursos para o Projecto de Empresas Ágeis / Virtuais para Produtos Complexos*, Tese de Doutoramento, Universidade do Minho, 2004, cap. 5

[3] Bastos, J., Azevedo, A. - *Empresas Virtuais - Novos paradigmas organizacionais*, I.S.E.P., www.dem.isep.ipp.pt/docentes/jab/files/EV.pdf

[4] Chapman,Stephen J. - *Programação em MATLAB para Engenheiros*, Thomson, 2006

[5] Choi, K.; Kim, D.; Doh, Y. - *Multi-agent-based task assignment system for virtual enterprises*, 2007, pag. 624-629

[6] Higham, D.J.; Higham, N.J. - *Matlab guide*. 2nd edition, SIAM, 2005

[7] Hsu, HP.; Hsu, HM. - *Systematic modeling and implementation of a resource planning system for virtual enterprise by Predicate/Transition net*, 2008, pag. 1841-1857

[8] Kothari, C.R. - *Research Methodology: Methods and Technics*, 2nd revised edition, New Age International Publishers, 2004

[9] MATLAB, http://cimss.ssec.wisc.edu/wxwise/class/aos340/spr00/whatismatlab.htm

[10] MATLAB, http://w3.impa.br/~zubelli/tutorial/node1.html

[11] MATLAB, http://www.mathworks.com/help/techdoc/ref/find.html

[12] Pires, A. - *Uma Reflexão sobre a Análise do Valor e o seu Posicionamento no Actual Panorama da Gestão da Qualidade*, Tese de Mestrado em Engenharia Mecânica, 2000 – Faculdade de Engenharia da Universidade do Porto

[13] Pires, A., Putnik, G., Ávila, P. – *The Potentialities of the Application of Value Analysis*, Proceedings of the 24th International Manufacturing Conference, pag. 745-751, Waterford, Ireland, 2007

[14] Pires, A., Putnik, G., Ávila, P. – *An Analysis about the Resources Selection Process in Agile/Virtual Enterprises*, in Goran D. Putnik and Paulo Ávila (Eds.) - Business Sustainability I, Management, Technology and Learning for Individuals, Organisations and Society in Turbulent Environments, U.Minho and I.S.E.P. (publishers), 2010, pag.196-202.

[15] Pithon, A. - *Projecto Organizacional para a Engenharia Concorrente no Âmbito das Empresas Virtuais*. Dissertação de Doutoramento em Engenharia de Produção e Sistemas, Universidade do Minho, Guimarães (2004)

[16] Putnik, G. - *Agile Manufacturing: 21st Century Manufacturing Strategy*, *BM_Virtual Enterprise Architecture Reference Model*. Gunasekaran A. (Ed.), Elsevier Science Publishers, p. 73-93.

[17] Putnik, G., Cunha, M.M., Sousa, R., Ávila, P. – "*BM_Virtual Enterprise: A Model for Dynamics and Virtuality*", em Putnik, G., Cunha, M.M. (Eds.) "*Virtual Enterprise Integration: Technological and Organizational Perspectives*", IDEA Group Publishing, Hershey, PA, USA, 2005, Chapter VI, pag. 124-143

[18] Schmidt,R. - *Busca e selecção de parceiros para Empresas Virtuais: Uma abordagem baseada em agentes móveis*. Dissertação para obtenção de grau de Mestre em Engenharia Eléctrica, Florianópolis, Julho de 2003

Anexo A. Outputs do programa

Número de plano de tarefas a analisar: 2

Para o plano de tarefas 1 indique número de tarefas

N.º de tarefas: 2

Para o plano de tarefas 2 indique número de tarefas

N.º de tarefas: 3

Plano de tarefas n.º 1

O número de recursos é igual para todas as tarefas? (s/n): s

Introduza o número de recursos candidatos às tarefas: 15

Plano de tarefas n.º 2

O número de recursos é igual para todas as tarefas? (s/n): s

Introduza o número de recursos candidatos às tarefas: 10

Introduza os valores de ponderação para os 3 sistemas em % (Se todos iguais prima enter):

Introduza os 5 valores de ponderação da qualidade em % (Se todos iguais prima enter):

Introduza os 5 valores de ponderação financeiros em % (Se todos iguais prima enter):

Introduza os 5 valores de ponderação das Sinergias em % (Se todos iguais prima enter):

Resultados da pré-selecção para o Plano de Tarefas 1

~~~~~~

Para a Tarefa 1

Referente ao nível 1 do algoritmo (sem avaliação do valor dos recursos)

Candidataram-se 15 recursos, dos quais foram aprovados 7 no nível 1 do algoritmo de pré-selecção
~~~~~~

Referente ao nível 2 do algoritmo (com avaliação do valor dos recursos)

6 recursos tiveram nota positiva no S.Q.

5 recursos tiveram nota positiva no S.F.

5 recursos tiveram nota positiva no S.S.

2 recursos tiveram nota positiva nos 3 sistemas

Recurso	V.S.Q.	V.S.F.	V.S.S.	C.P.
10	5.7163	6.5760	6.8497	6.3807
3	6.2645	5.1981	6.3608	5.9411

~~~~~~

Para a Tarefa 2

Referente ao nível 1 do algoritmo (sem avaliação do valor dos recursos)

Candidataram-se 15 recursos, dos quais foram aprovados 11 no nível 1 do algoritmo de pré-selecção

Referente ao nível 2 do algoritmo (com avaliação do valor dos recursos)

10 recursos tiveram nota positiva no S.Q.

7 recursos tiveram nota positiva no S.F.

9 recursos tiveram nota positiva no S.S.

5 recursos tiveram nota positiva nos 3 sistemas

Recurso	V.S.Q.	V.S.F.	V.S.S.	C.P.
14	7.1231	7.2016	5.4074	6.5774
10	6.6560	6.0357	6.4311	6.3743
3	6.6938	6.9126	5.5072	6.3712
6	5.3152	7.0992	5.4173	5.9439
8	6.0311	5.8862	5.1322	5.6832

Optimization terminated.

~~~~~~

Resultados da selecção final para o Plano de Tarefas 1

N.º de tarefas: 2

N.º inicial de recursos candidatos: 15

k minimo do nível 1: 7

Sem análise de valor

Sistema de recursos: R1,6; R2,10;

Custo do sistema: 12.0788

CP: 6.0846

CP R1,6= 3.5843;

CP R2,10= 2.5002;

CT: 5.9943

CT 1->2= 5.9943

Valor do sistema: 12.3900

Valor R1,6= 6.0157;

Valor R2,10= 6.3743;

Nº de recursos seleccionados com classificação negativa: 0

Foi encontrada a solução óptima

N.º de iterações efectuadas: 64

Tempo da simulação = 0 horas,0 minutos e 3 segundos

Optimization terminated.

~~~~~~

N.º inicial de recursos candidatos: 15

k minimo do nível 1: 7

k minimo do nível 2: 2

Com análise de valor
~~~~~~

Sistema de recursos: R1,10; R2,10;

Custo do sistema: 13.8787

CP: 9.4341

CP R1,10= 6.9338;

CP R2,10= 2.5002;

CT: 4.4446

CT 1->2= 4.4446

Valor do sistema: 12.7549

Valor R1,10= 6.3807;

Valor R2,10= 6.3743;

Foi encontrada a solução óptima

N.º de iterações efectuadas: 5

Tempo da simulação = 0 horas,0 minutos e 0 segundos

~~~~~~

Tempo total da simulação = 0 horas,0 minutos e 3 segundos

Resultados da pré-selecção para o Plano de Tarefas 2

~~~~~~

Para a Tarefa 1

Referente ao nível 1 do algoritmo (sem avaliação do valor dos recursos)

Candidataram-se 10 recursos, dos quais foram aprovados 8 no nível 1 do algoritmo de pré-selecção

Referente ao nível 2 do algoritmo (com avaliação do valor dos recursos)

7 recursos tiveram nota positiva no S.Q.

7 recursos tiveram nota positiva no S.F.

5 recursos tiveram nota positiva no S.S.

4 recursos tiveram nota positiva nos 3 sistemas

Recurso	V.S.Q.	V.S.F.	V.S.S.	C.P.
8	6.8253	7.0106	5.4204	6.4188
3	5.3680	6.6322	6.5880	6.1961
7	5.1985	7.2654	5.8175	6.0938
4	5.4796	6.3647	5.9122	5.9188

~~~~~~

Para a Tarefa 2

Referente ao nível 1 do algoritmo (sem avaliação do valor dos recursos)

Candidataram-se 10 recursos, dos quais foram aprovados 10 no nível 1 do algoritmo de pré-selecção

Referente ao nível 2 do algoritmo (com avaliação do valor dos recursos)

10 recursos tiveram nota positiva no S.Q.

10 recursos tiveram nota positiva no S.F.

8 recursos tiveram nota positiva no S.S.

8 recursos tiveram nota positiva nos 3 sistemas

Recurso	V.S.Q.	V.S.F.	V.S.S.	C.P.
7	6.9561	6.2799	7.7712	7.0024
6	6.3860	6.8931	5.9445	6.4079
5	5.5024	7.2279	6.3785	6.3696
4	5.1339	6.8271	6.6543	6.2051
9	6.4255	6.4771	5.5995	6.1674
8	7.0226	6.2242	5.2225	6.1565
3	6.0653	6.1474	5.9510	6.0545
2	5.9729	5.8555	5.3195	5.7160
~~~~~~

~~~~~~

Para a Tarefa 3

Referente ao nível 1 do algoritmo (sem avaliação do valor dos recursos)

Candidataram-se 10 recursos, dos quais foram aprovados 5 no nível 1 do algoritmo de pré-selecção

Referente ao nível 2 do algoritmo (com avaliação do valor dos recursos)

4 recursos tiveram nota positiva no S.Q.

4 recursos tiveram nota positiva no S.F.

5 recursos tiveram nota positiva no S.S.

4 recursos tiveram nota positiva nos 3 sistemas

Recurso	V.S.Q.	V.S.F.	V.S.S.	C.P.
9	6.2710	5.5223	7.1928	6.3287
7	5.5704	6.7397	6.1132	6.1411
4	5.7111	5.9921	6.5087	6.0706
3	5.4531	6.2870	5.0912	5.6104

Optimization terminated.

Resultados da selecção final para o Plano de Tarefas 2

N.º de tarefas: 3

N.º inicial de recursos candidatos: 10

k minimo do nível 1: 5

Sem análise de valor

Sistema de recursos: R1,4; R2,5; R3,9;

Custo do sistema: 22.8070

CP: 12.7298

CP R1,4= 4.2071;
~~~~~~

CP R2,5= 3.688;

CP R3,9= 4.8347;

CT: 10.0771

CT 1->2= 6.7169

CT 2->3= 3.3602

Valor do sistema: 18.6171

Valor R1,4= 5.9188;

Valor R2,5= 6.3696;

Valor R3,9= 6.3287;

Nº de recursos seleccionados com classificação negativa: 0

Foi encontrada a solução óptima

N.º de iterações efectuadas: 128

Tempo da simulação = 0 horas,0 minutos e 1 segundos

Optimization terminated.

~~~~~~

N.º inicial de recursos candidatos: 10

k minimo do nível 1: 5

k minimo do nível 2: 4

Com análise de valor

Sistema de recursos: R1,4; R2,5; R3,9;

Custo do sistema: 22.8070

CP: 12.7298

CP R1,4= 4.2071;

CP R2,5= 3.688;

CP R3,9= 4.8347;
~~~~~~

CT: 10.0771

CT 1->2= 6.7169

CT 2->3= 3.3602

Valor do sistema: 18.6171

Valor R1,4= 5.9188;

Valor R2,5= 6.3696;

Valor R3,9= 6.3287;

Foi encontrada a solução óptima

N.º de iterações efectuadas: 94

Tempo da simulação = 0 horas,0 minutos e 0 segundos

~~~~~~

Tempo total da simulação = 0 horas,0 minutos e 1 segundos
~~~~~~

Anexo B. Programa em código MATLAB

Módulo base

```
%% Dados de entrada

clc
%Vai ser pedido vector dos minimos admissiveis para a filtragem do nivel
1

%Sma_1=input('Introduza os valores minimos na forma de vector:\n');
Sma_1=[1 1 1 1];

%Vai ser pedido o vector dos minimos admissiveis para a filtragem da
matriz
%final
%Sma_2=input('Introduza os valores minimos do nivel 2 na forma de
vector:\n');
Sma_2=[5 5 5];

nPT=input('Número de plano de tarefas a analisar: ');
nTare=zeros(1,nPT); %cria o vector que vai guardar o numero de tarefas de
cada PT

for ii=1:nPT
    fprintf('\nPara o plano de tarefas %d indique número de tarefas',ii);
    nTare(ii)=input('\nN.º de tarefas: ');
end

%clc

Xc=cell(1,nPT); %cria o vector de matrizes para guardar o numero de
recursos candidatos de cada tarefa
for ii=1:nPT
    perg=[];
    fprintf('\nPlano de tarefas n.º %d\n',ii);
    if nTare(ii) > 1
        perg=input('O número de recursos é igual para todas as tarefas?
(s/n): ','s');
    end
    if perg== 's'
        Xc{ii}(1)=input('\nIntroduza o número de recursos candidatos às
tarefas: ');
        for jj=1:nTare(ii)
            Xc{ii}(jj)=Xc{ii}(1);
        end
    else
        for jj=1:nTare(ii)
            fprintf('\nTarefa número %d', jj);
            Xc{ii}(jj)=input('\nIntroduza o número de recursos candidatos
à tarefa: ');
        end
    end
    clear perg
end
```

```
%% Pede os valores de ponderação dos 3 parâmetros de avaliação

% ponderações finais
 PondF=input('\nIntroduza os valores de ponderação para os 3 sistemas em
% (Se todos iguais prima enter):\n');
 pondq=input('Introduza os 5 valores de ponderação da qualidade em % (Se
todos iguais prima enter):\n');
 pondf=input('Introduza os 5 valores de ponderação financeiros em % (Se
todos iguais prima enter):\n');
 ponds=input('Introduza os 5 valores de ponderação das Sinergias em % (Se
todos iguais prima enter):\n');


%% Verifica se a soma dos parâmetros de ponderação passa dos 100%. Se
passar pede para introduzir outra vez os valores

while sum(pondq)>100 || sum(pondq)<100 && sum(pondq)>0
      pondq=input('\nIntroduza novamente os 5 valores de ponderação da
qualidade em % (Se todos iguais prima enter):\n');
end

while sum(pondf)>100 || sum(pondf)<100 && sum(pondf)>0
      pondf=input('\nIntroduza novamente os 5 valores de ponderação
financeiros em % (Se todos iguais prima enter):\n');
end

while sum(ponds)>100 || sum(ponds)<100 && sum(ponds)>0
      ponds=input('\nIntroduza novamente os 5 valores de ponderação das
Sinergias em % (Se todos iguais prima enter):\n');
end

while sum(PondF)>100 || sum(PondF)<100 && sum(PondF)>0
      PondF=input('\nIntroduza novamente os valores de ponderação dos 3
sistemas % (Se todos iguais prima enter):\n');
end

%% Trabalha valores de ponderação

%Ponderação da qualidade
if isempty(pondq)
   pondq=100/(5*100); %(5*100) para dar o valor em percentagem (0,2)
else
   pondq=pondq/100;
end

%Ponderação Financeira
if isempty(pondf)
   pondf=100/(5*100); %(5*100) para dar o valor em percentagem (0,2)
else
   pondf=pondf/100;
end

%Ponderação Sinergias
if isempty(ponds)
   ponds=100/(5*100); %(5*100) para dar o valor em percentagem (0,2)
else
   ponds=ponds/100;
end
```

```
%Ponderação final
if isempty(PondF)
   PondF=100/(3*100); %(3*100) para dar o valor em percentagem (0,333)
else
   PondF=PondF/100;
end

%% Definição dos limites superiores e inferiores dos CP e CT e criação do
vector de matrizes que vai guardar os resultados

flag=0;
Flag_ni2=0;
valor=cell(5,nPT); %vector de matrizes que vai armazenar a matriz dos
recursos seleccionados com o valor de cada um (soma dos 3S, o valor do
sistema, o vector dos CP. e CT. e o custo total do sistema)

valor2=cell(5,nPT);
%% Análise e manipulação dos dados

for ii=1:nPT
    %chamada da função de pré-selecção, retornando o DRP e o vector dos
CP
    [DRP, CP]=pre_Selec(nTare(ii), Xc{ii}, Sma_1, pondq, pondf, ponds,
PondF, Sma_2,ii);

%% Chama a 1ª vez a função de selecção final para cálculo sem análise de
valor
    k_min1=size(DRP{4,1},1);

    [x1, fx1, exitflag1, output1, CT]=Selec_BB(nTare(ii),k_min1,CP); %CP
vector com nTare*k(n.º de recursos) %CT (nTare-1)*k*k

    Flag_ni2=0; %flag para saber se vai apresentar dados com/sem análise
de valor na função Apr_res_selec

    locali=find(x1==1); %vector com a localização dos recursos
seleccionados

    for jj=1:nTare(ii)
        posi=locali(jj)-k_min1*(jj-1); %localização do recurso para cada
tarefa
        valor{1,ii}(jj,1)=DRP{4,jj}(posi,1); %copia o n.º do recurso
        valor{1,ii}(jj,2)=DRP{4,jj}(posi,5); %copia a classificação final
do recurso
        valor{1,ii}(jj,3)=DRP{4,jj}(posi,6); %copia os custos de
processamento do recurso
    end

    valor{2,ii}=0;
    for jj=1:nTare(ii) %para fazer o cálculo do valor do sistema
        valor{2,ii}=valor{2,ii}+valor{1,ii}(jj,2);
    end

    valor{3,ii}=CP;
    valor{4,ii}=CT;
    valor{5,ii}=fx1;
```

```
Apr_res_selec(valor,locali,nTare(ii),ii,Xc{ii}(1),Flag_ni2,output1,exitfl
ag1,k_min1);

  %% Se o k_min2 for diferente de zero é que vai encontrar o novo vector
dos CT e chamar a função de selecção pela 2ª vez
    k_min2=size(DRP{6,1},1);

    if k_min2==0
        fprintf('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~');
        fprintf('\nNão existem recursos para a realização dos cálculos
com análise de valor\n');
    else
%% Altera o vector de CT (os custos de transporte vão ser analisados aos
pares de tarefas, tarefa 1->2, tarefa 2->3, tarefa 3->4...)

        CT2=[];
        for jj=1:nTare(ii)-1
            m=reshape(CT(1+k_min1*k_min1*(jj-
1):jj*k_min1*k_min1),size(DRP{4,1},1),size(DRP{4,1},1))'; %transforma o
vector CT numa matriz de k*k

%A ' é para que preencha a matriz linha a linha e não coluna a coluna
        % Para relações entre tarefas impares e pares
            if flag==0
                for kk=jj:jj+1
                    a=DRP{4,kk}(:,1)';              %copia os n.º dos
recursos aprovados no 1º nível
                    b=DRP{6,kk}(:,1)';              %copia os n.º dos
recursos aprovados no 2º nível
% Com os 2 vectores de recursos criados é preciso encontrar as posições
dos que foram eliminados de um nivel para outro
                    ini=1;
                    pos=zeros(1,size(a,2));
                    for mm=1:size(b,2)              %para o comprimento
de b
                        for ll=ini:size(a,2)        %vai percorrer o
vector a para comparação
                            if b(mm)~=a(ll)
                                pos(ll)=0;          %se o valor for
diferente preenche com zero
                            else
                                pos(ll)=1;          %se o valor for igual
preenche com 1
                                ini=ll+1;           %incrementa 1 para
passar a frente um valor
                                break               %faz o break para
mudar de numero de b a comparar
                            end
                        end
                    end

                    pos2=find(pos==0);                %guarda as
posições em que os recursos são diferentes
% Eliminação de linhas e colunas na matriz dos custos
                    if mod(kk,2)==1                 %para o n.º de tarefa
ímpar (1 3 5...) preenche as linhas dos recursos que foram eliminados de
                        for mm=1:size(pos2,2)           % um nível para
o outro com zeros
                            m(pos2(mm),:)=zeros;
                        end
```

```
                    else                                %para o n.º de tarefa
par (2 4 6...)preenche as linhas dos recursos eliminados com zeros
                        for mm=1:size(pos2,2)
                            m(:,pos2(mm))=zeros;
                        end
                    end
                    clear pos;          %elimina o pos para que da
próxima vez não escreva por cima e crie erros
                end
            flag=1;

        % Para relações entre tarefas pares e impares
            else
                for kk=jj:jj+1
                    a=DRP{4,kk}(:,1)';               %copia os n.º de
recursos aprovados no 1º nível
                    b=DRP{6,kk}(:,1)';               %copia os n.º de
recursos aprovados no 2º nível
% Com os 2 vectores de recursos criados é preciso encontrar as posições
dos que foram eliminados de um nivel para outro
                    ini=1;
                    pos=zeros(1,size(a,2));
                    for mm=1:size(b,2)              %para o comprimento
de b
                        for ll=ini:size(a,2)        %vai percorrer o
vector a para comparação
                            if b(mm)~=a(ll)
                                pos(ll)=0;          %se o valor for
diferente preenche com zero
                            else
                                pos(ll)=1;          %se o valor for igual
preenche com 1
                                ini=ll+1;           %incrementa 1 para
passar a frente um valor
                                break               %faz o break para
mudar de numero de b a comparar
                            end
                        end
                    end

                    pos2=find(pos==0);                  %guarda as
posições em que os recursos são diferentes
% Eliminação de linhas e colunas na matriz dos custos
                    if mod(kk,2)==0                 %para o n.º de tarefa
par (2 4 6...) preenche as linhas dos recursos que foram eliminados de
                        for mm=1:size(pos2,2)            % um nível para
o outro com zeros
                            m(pos2(mm),:)=zeros;
                        end
                    else                                %para o n.º de tarefa
impar (3 5 7...)preenche as linhas dos recursos eliminados com zeros
                        for mm=1:size(pos2,2)
                            m(:,pos2(mm))=zeros;
                        end
                    end
                    clear pos;          %elimina o pos para que da
próxima vez não escreva por cima e crie erros
                end
            flag=0;
            end
```

```
            m=m';   %torna as linhas em colunas pq o find faz a busca por
colunas... e é necessário manter a ordem dos valores como se fossem
linhas
            x=find(m~=0)';
            Ct2=zeros(1,size(x,2));
            for kk=1:size(x,2)
                Ct2(kk)=m(x(kk));
            end
        CT2=[CT2, Ct2];
        end
%% Altera o valor de CP
        CP2=[];
        for jj=1:nTare(ii)
            Cp2=DRP{6,jj}(:,6)';

            CP2=[CP2, Cp2];
        end

%% Chama a 2ª vez a função de selecção final para cálculo com análise de
valor

        [x2, fx2, exitflag2, output2,
CT2]=Selec_BB(nTare(ii),k_min2,CP2,CT2);

        Flag_ni2=1;

        locali2=find(x2==1); %vector com a localização dos recursos
seleccionados

        for jj=1:nTare(ii)
            pos=locali2(jj)-k_min2*(jj-1); %localização do recurso para
cada tarefa
            valor2{1,ii}(jj,1)=DRP{6,jj}(pos,1); %copia o n.º do recurso
            valor2{1,ii}(jj,2)=DRP{6,jj}(pos,5); %copia a classificação
final do recurso
            valor2{1,ii}(jj,3)=DRP{6,jj}(pos,6); %copia os custos de
processamento do recurso
        end

        valor2{2,ii}=0;
        for jj=1:nTare(ii) %para fazer o cálculo do valor do sistema
            valor2{2,ii}=valor2{2,ii}+valor2{1,ii}(jj,2);
        end

        valor2{3,ii}=CP2;
        valor2{4,ii}=CT2;
        valor2{5,ii}=fx2;


Apr_res_selec(valor2,locali2,nTare(ii),ii,Xc{ii}(1),Flag_ni2,output2,exit
flag2,k_min1,k_min2);

%% Apresentação do tempo total de simulação (armazenado nas variaveis
outputX.time)
        segundos=output1.time+output2.time;
        h=floor(segundos/3600);         %a parte inteira sao as horas
        segundos2=segundos-(h*3600);    %multiplica-se as horas por 3600
e subtrai-se ao tempo inicial (fica-se com os minutos e segundos)
        min=floor(segundos2/60);        %a parte inteira são os minutos
```

```
        seg=round(segundos2-(min*60));

        fprintf('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~');
        fprintf('\nTempo total da simulação = %d horas,%d minutos e %d
segundos\n',h,min,seg);

    end
%% Muda o nome das variáveis

    varname=['x1_PT' num2str(ii)]; %altera o nome do vector dos recursos
escolhidos para cada tarefa
    assignin('base',varname,x1);
    clear x1;

    varname=['x2_PT' num2str(ii)]; %altera o nome do vector dos recursos
escolhidos para cada tarefa
    assignin('base',varname,x2);
    clear x2;

    varname=['DRP' num2str(ii)]; %altera o nome do DRP
    assignin('base',varname,DRP);
    clear DRP;

end
```

Módulo pré_selec

```
function [DRP, CP]=pre_Selec(nTare, Xc, Sma_1, pondq, pondf, ponds,
PondF, Sma_2,pp)
%Function pre_Selec.m
%Função de pré-selecção de recursos

% inputs
%   nTare - n.º de tarefas do PT
%   Xc - n.º de recursos candidatos a cada tarefa
%   Sma_1 - Vector dos mínimos de aceitação admissíveis para o 1º nível
%   Sma_2 - Vector dos mínimos de aceitação admissíveis para o 2º nível
%   pond - Vectores com os valores de ponderações de cada sistema e Final
%   pp - n.º do Plano de Tarefas em análise
% output
%   DRP - matriz de matrizes contendo os recursos que passaram as
diversas
% fases de avaliação


%% Nivel 1 do algoritmo

fprintf('\n-------------------------------------------------------------------
-------------------------\n');
fprintf('-------------------------------------------------------------------
-----------------------\n');
fprintf('Resultados da pré-selecção para o Plano de Tarefas %d\n',pp);
%% Preenchimento aleatório da matriz dos recursos candidatos
for ii=1:1
    DRP=cell(6,nTare);
    for jj=1:nTare
```

```
        Drp=zeros(Xc(jj),21);% A matriz de recursos vai ser preenchida
aleatoriamente
        for kk=1:Xc(jj)
            Drp(kk,1)=kk;
            Drp(kk,2:5)=binornd(1, 0.90,1,4);
            Drp(kk,6:20)=6+2.*randn(1,15); %6 é a média + o desvio padrão
de 2
            Drp(kk,21)=6+2.*randn(1); %custos de processamento
        end

%% garantir que os valores introduzidos estão dentro do intervalo
pretendido
%[0 - 10]

        for kk=1:size(Drp,1)
            for mm=6:(size(Drp,2)-1)
                if Drp(kk,mm)<0
                   Drp(kk,mm)=0;
                elseif Drp(kk,mm)>10
                    Drp(kk,mm)=10;
                end
            end
            if Drp(kk,21)<0
                Drp(kk,21)=abs(Drp(kk,21));
            end
        end



        %% contador do numero de linhas eliminadas na matriz das
aprovações
        cont=0;
        flag=0; %flag para se saber se alguma linha foi eliminada.
        %se sim, o valor de ii tem de ser decrementado aquando da copia
do
        %valor

        %% Avaliação nivel 1 - Comparação entre as matrizes e criação de
matriz com dados 2° nivel

        Drp_apr = zeros(Xc(jj),17); %inicia a matriz com zeros para ser
mais rápido
        for kk=1:Xc(jj)
            if flag == 0
               Drp_apr(kk,1)=Drp(kk,1);
               if Drp(kk,2)== Sma_1(1) && Drp(kk,3)== Sma_1(2) &&
Drp(kk,4)== Sma_1(3) && Drp(kk,5)== Sma_1(4) %Se os mínimos admissíveis
forem atingidos
                  Drp_apr(kk,2:17)= Drp(kk,6:21); %copia as restantes
colunas da matriz inicial
               else
                  flag = 1;
                  Drp_apr(kk,:)=[]; %Se os mínimos não forem atingidos
elimina a linha referente ao produto q ñ cumpre os req.
               end
            else
                Drp_apr(kk-cont-1,1)=Drp(kk,1); %A mesma coisa de cima
mas contemplando o caso de já ter sido eliminada alguma linha
               if Drp(kk,2)== Sma_1(1) && Drp(kk,3)== Sma_1(2) &&
Drp(kk,4)== Sma_1(3) && Drp(kk,5)== Sma_1(4)
                  Drp_apr(kk-cont-1,2:17)=Drp(kk,6:21);
```

```
                else
                    cont=cont+1;  % conta o numero de linhas que foi
eliminado
                    Drp_apr(kk-cont,:)=[];
                end
            end
        end


        if size(Drp_apr,1)==0
           fprintf('Não existem recursos com classificação aceitável.
Considere a utilização de graus de flexibilidade\n');
        end

%% Nível 2 do algoritmo

%% Cálculo das classificações de cada recurso e armazenamento na última
coluna da respectiva matriz

        Drp_final=zeros(size(Drp_apr,1),6); %Inicialização da matriz
final

%% Ponderação qualidade

        if size(pondq,2)==1
           for kk=1:size(Drp_apr,1)
               Drp_final(kk,1)=Drp_apr(kk,1);
               for mm=2:6
                   Drp_final(kk,2)=Drp_final(kk,2)+pondq*Drp_apr(kk,mm);
               end
           end
        else
            for kk=1:size(Drp_apr,1)%contr
                Drp_final(kk,1)=Drp_apr(kk,1);
                for mm=2:6
                    Drp_final(kk,2)=Drp_final(kk,2)+pondq(mm-
1)*Drp_apr(kk,mm);
                end
            end
        end

%% Ponderação Financeiro

        if size(pondf,2)==1
            for kk=1:size(Drp_apr,1)
                for mm=7:11
                    Drp_final(kk,3)=Drp_final(kk,3)+pondf*Drp_apr(kk,mm);
                end
            end
        else
            for kk=1:size(Drp_apr,1)
                for mm=7:11
                    Drp_final(kk,3)=Drp_final(kk,3)+pondf(mm-
6)*Drp_apr(kk,mm);
                end
            end
        end

%% Ponderação Sinergias
```

```
        if size(ponds,2)==1
            for kk=1:size(Drp_apr,1)
                for mm=12:16
                    Drp_final(kk,4)=Drp_final(kk,4)+ponds*Drp_apr(kk,mm);
                end
            end
        else
            for kk=1:size(Drp_apr,1)
                for mm=12:16
                    Drp_final(kk,4)=Drp_final(kk,4)+ponds(mm-
11)*Drp_apr(kk,mm);
                end
            end
        end

%% Ponderação dos 3 sistemas

        if size(PondF,2)==1
            for kk=1:size(Drp_apr,1)
                for mm=2:4

Drp_final(kk,5)=Drp_final(kk,5)+PondF*Drp_final(kk,mm);
                    Drp_final(kk,6)=Drp_apr(kk,17); %para copiar os
custos de processamento (podia estar em qualquer sitio. Só ficou aqui
para aproveitar o ciclo)
                end
            end
        else
            for kk=1:size(Drp_apr,1)
                for mm=2:4
                    Drp_final(kk,5)=Drp_final(kk,5)+PondF(mm-
1)*Drp_final(kk,mm);
                    Drp_final(kk,6)=Drp_apr(kk,17); %para copiar os
custos de processamento
                end
            end
        end

        DRP{3,jj}=sortrows(Drp_final,-5);

%% Para contar o numero de recursos com classificação positiva em cada
sistema

        npos_SQ=0; npos_SF=0; npos_SS=0; npos_3S=0;
        for kk=1:size(Drp_apr,1)
            if Drp_final(kk,2)>=5
                npos_SQ=npos_SQ+1; %número de recursos com classificação
no S.Q. positiva >=5
            end
            if Drp_final(kk,3)>=5
                npos_SF=npos_SF+1; %número de recursos com classificação
no S.F. positiva >=5
            end
            if Drp_final(kk,4)>=5
                npos_SS=npos_SS+1; %número de recursos com classificação
no S.S. positiva >=5
            end
            if Drp_final(kk,2)>=5 && Drp_final(kk,3)>=5 &&
Drp_final(kk,4)>=5
```

```
                npos_3S=npos_3S+1; %número de recursos com classificação
nos 3 sistemas positiva >=5
            end
        end

%% elimina os recursos que não tenham classificação positiva nos 3
sistemas (ou classificação inferior ao definido em Sma_2)
        for kk=1:size(Drp_apr,1)
            for mm=2:4
                if Drp_final(kk,mm)< Sma_2(mm-1)
                    Drp_final(kk,:)= zeros; %Preenche as linhas que não
correspondem à condição com zeros
                    break
                end
            end
        end

        Drp_final(~all(Drp_final,2),:)=[];  %Todas as linhas que tenham
zeros são eliminadas

%% Organização por ordem decrescente da classificação final

        Drp_final=sortrows(Drp_final,-5);

%% Apresentação dos resultados
        fprintf('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~');
        fprintf('\nPara a Tarefa %d\n',jj);
        fprintf('\n  Referente ao nível 1 do algoritmo (sem avaliação do
valor dos recursos)\n');
        fprintf('\n\tCandidataram-se %d recursos, dos quais foram
aprovados %d no nível 1 do algoritmo de pré-selecção', Xc(jj),
size(Drp_apr,1));
        fprintf('\n\n  Referente ao nível 2 do algoritmo (com avaliação
do valor dos recursos)\n');
        fprintf('\n\t%d recursos tiveram nota positiva no S.Q.',
npos_SQ);
        fprintf('\n\t%d recursos tiveram nota positiva no S.F.',
npos_SF);
        fprintf('\n\t%d recursos tiveram nota positiva no S.S.',
npos_SS);
        fprintf('\n\t%d recursos tiveram nota positiva nos 3 sistemas\n',
npos_3S);
        if npos_3S~=0
           fprintf('\n  Recurso    V.S.Q.     V.S.F.     V.S.S.
C.P.\n');
            for kk=1:size(Drp_final,1)
                fprintf('    %3d     %7.4f    %7.4f    %7.4f    %7.4f\n',
Drp_final(kk,1),Drp_final(kk,2),Drp_final(kk,3),Drp_final(kk,4),Drp_final
(kk,5));
            end
        end
        DRP{1,jj}=Drp;
        DRP{2,jj}=Drp_apr;
        DRP{5,jj}=Drp_final;

    end

    %% No final de todas as tarefas analisadas, considera-se o mesmo n.º
de recursos aprovados no nível 1 em todas as tarefas -> menor número de
todas
```

```
    if jj==nTare
        DIM=zeros(1,nTare); %cria vector onde vão ser guardadas as
dimensões das matrizes finais
        for kk=1:nTare
            DIM(kk)=size(DRP{3,kk},1); %preenche o vector com os valores
das dimensões
            DRP{4,kk}=DRP{3,kk}; %copia a matriz final para a celula 4
        end
        DIM_min=min(DIM); %atribui o valor mais pequeno do vector à
variavel
        for kk=1:nTare
            for mm=DIM_min+1:DIM(kk) %do valor mais pequeno +1 até ao
final da dimensao da matriz
                DRP{4,kk}(mm,:)=zeros; %preenche com zeros
            end
            DRP{4,kk}(~all(DRP{4,kk},2),:)=[]; %apaga todas as linhas com
zeros
        end
    end

%% É preciso juntar todos os custos de processamento dos recursos que
passam o 1º nível tendo em conta o k_min (4ª linha do DRP)

    CP=[];
    for jj=1:nTare
        Cp=DRP{4,jj}(:,6)'; %6 corresponde à última coluna da matriz
(custos)
        CP=[CP, Cp];
    end

%% No final de todas as tarefas analisadas, considera-se o mesmo n.º de
recursos aprovados no nível 2 em todas as tarefas -> menor número de
todas
    if jj==nTare
        DIM2=zeros(1,nTare); %cria vector onde vão ser guardadas as
dimensões das matrizes finais
        for kk=1:nTare
            DIM2(kk)=size(DRP{5,kk},1); %preenche o vector com os valores
das dimensões
            DRP{6,kk}=DRP{5,kk}; %copia a matriz final para a celula 4
        end
        DIM2_min=min(DIM2); %atribui o valor mais pequeno do vector à
variavel
        for kk=1:nTare
            for mm=DIM2_min+1:DIM2(kk) %do valor mais pequeno +1 até ao
final da dimensao da matriz
                DRP{6,kk}(mm,:)=zeros; %preenche com zeros
            end
            DRP{6,kk}(~all(DRP{6,kk},2),:)=[]; %apaga todas as linhas com
zeros
        end
    end
end
```

Módulo Apr_res_selec

```
Function
Apr_res_selec(valor,locali,nTare,pp,Xc,Flag_ni2,output,exitflag,k_min1,va
rargin)
%Função de apresentação dos resultados da selecção final (C/ ou S/ AV)
if size(varargin,2)==1
    k_min2= varargin{1};
end

if Flag_ni2==0
   fprintf('-------------------------------------------------------------
-------------------------\n');
   fprintf('Resultados da selecção final para o Plano de Tarefas %d',pp);
   fprintf('\n\nN.º de tarefas: %d',nTare);
   fprintf('\nN.º inicial de recursos candidatos: %d',Xc); %só funciona
correctamente se o n.º de recursos candidatos for igual em todas as
tarefas
   fprintf('\nk minimo do nível 1: %d', k_min1);
   fprintf('\n\nSem análise de valor');
else
   fprintf('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~');
   fprintf('\nN.º inicial de recursos candidatos: %d',Xc); %só funciona
correctamente se o n.º de recursos candidatos for igual em todas as
tarefas
   fprintf('\nk minimo do nível 1: %d', k_min1);
   fprintf('\nk minimo do nível 2: %d', k_min2);
   fprintf('\n\nCom análise de valor');
end
   fprintf('\n\n\tSistema de recursos: ');

  %% Para a apresentação dos recursos seleccionados (percorre a matriz
dos seleccionados e concatena o numero da tarefa (linha) ao n.º do
recurso escolhido para essa tarefa
   for kk=1:size(valor{1,pp},1)
       varname=['R' num2str(kk) ',' num2str(valor{1,pp}(kk,1)) '; '];
       fprintf(varname); %não põe \n no final
   end

  %% Apresentação dos custos do sistema
   fprintf('\n\n\tCusto do sistema: %.4f', valor{5,pp});

  %% Custos de processamento
   fprintf('\n\t\tCP: %.4f\n',sum(valor{1,pp}(:,3)));

  %% Custos de processamento individuais
   for kk=1:size(valor{1,pp},1)
       varname=['CP R' num2str(kk) ',' num2str(valor{1,pp}(kk,1)) '= '
num2str(valor{1,pp}(kk,3)) ';'];
       fprintf('\t\t\t');
       disp(varname); %põe \n no final
   end

  %% Custos de transporte
   fprintf('\n\t\tCT: %.4f\n',valor{5,pp}-sum(valor{1,pp}(:,3)));

  %% Custos de transporte individuais
   for mm=1:nTare-1
       varname=['CT ' num2str(mm) '->' num2str(mm+1) '= '
num2str(valor{4,pp}(locali(nTare+mm)-size(valor{3,pp},2)))];
```

```
        fprintf('\t\t\t');
        disp(varname);
    end

   %% Apresentação do valor do sistema
    fprintf('\n\tValor do sistema: %.4f\n', valor{2,pp});

   %% Apresentação do valor individual de cada recurso seleccionado
    for kk=1:size(valor{1,pp},1)
        varname=['Valor R' num2str(kk) ',' num2str(valor{1,pp}(kk,1)) '= '
num2str(valor{1,pp}(kk,2)) ';'];
        fprintf('\t\t');
        disp(varname);
    end

    %% Apresentação do número de recursos com classificação abaixo de 5
if Flag_ni2==0
    x=0;
    for kk=1:size(valor{1,pp},1)      %dos recursos seleccionados vai
contar os que têm class. negativa
        if valor{1,pp}(kk,2) < 5
            x=x+1;
        end
    end
    fprintf('\nN° de recursos seleccionados com classificação negativa:
%d\n',x);
end
    %% Apresentação dos dados da estrutura de OUTPUT (tempo, iterações e
flag)
    %Flag
     if exitflag == 1
         fprintf('\nFoi encontrada a solução óptima\n');
     elseif exitflag == 0
         fprintf('\nO número de iterações foi excedido\n');
     elseif exitflag == -2
         fprintf('\nNão foi encontrada solução possível\n');
     elseif exitflag == -4
         fprintf('\nO número de "nodes" foi excedido\n');
     elseif exitflag == -5
         fprintf('\nO tempo de execução foi excedido\n');
     elseif exitflag == -6
         fprintf('\nO número de iterações realizadas sobre o mesmo "node"
foi excedido\n');
     end

    %Iterações
    fprintf('\nN.° de iterações efectuadas: %d \n',output.iterations);

    %Tempo
     segundos=output.time;
     h=floor(segundos/3600);          %a parte inteira sao as horas
     segundos2=segundos-(h*3600);     %multiplica-se as horas por 3600 e
subtrai-se ao tempo inicial (fica-se com os minutos e segundos)
     min=floor(segundos2/60);         %a parte inteira são os minutos
     seg=round(segundos2-(min*60));

     fprintf('\nTempo da simulação = %d horas,%d minutos e %d
segundos\n',h,min,seg);

end
```