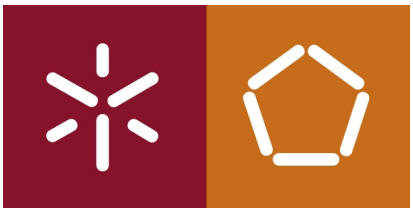


Universidade do Minho

Licenciatura em Engenharia Informática

Inteligência Artificial - Relatório Fase 1

Grupo 30

Pedro Aquino Martins de Araújo (A90614)
Ricardo Miguel Santos Gomes (A93785)
Rui Pedro Gomes Coelho (A58898)
Vasco Baptista Moreno (A94194)

dezembro, 2021

Capítulo 1

Resumo

O presente relatório pretende apoiar e descrever o trabalho desenvolvido na Fase 1 do trabalho proposto na Unidade Curricular de Inteligência Artificial, cujo objetivo central incidiu na modelação de um sistema de distribuição de encomendas.

Para o efeito foi usada a programação em lógica PROLOG que integrou uma base de conhecimento e um conjunto de predicados que permitem modelar diversas funcionalidades associadas com a logística do universo em estudo. Como tal, no decurso do relatório, serão dadas a conhecer as estratégias de modelação adotadas pelo grupo para representar corretamente o universo da logística de distribuição de encomendas.

Conteúdo

1 Resumo **2**

2 Introdução **7**

3 Preliminares **8**

4 Descrição do Trabalho e Análise de Resultados **9**

4.1 Base de Conhecimento 9

4.2 Queries 11

5 Conclusões e Sugestões **16**

6 Bibliografia **17**

A Predicados auxiliares **18**

Lista de Figuras

4.1 Base de Conhecimento: entrega. 10

4.2 Base de Conhecimento: estafetaRanking. 10

4.3 Base de Conhecimento: morada. 10

4.4 Query 1. 11

4.5 Teste query 1. 11

4.6 Query 2. 11

4.7 Teste query 2. 12

4.8 Query 3. 12

4.9 Teste query 3. 12

4.10 Query 4. 12

4.11 Teste query 4. 12

4.12 Query: 5. 13

4.13 Teste: query 5. 13

4.14 Query: 6. 13

4.15 Teste query 6. 13

4.16 Query: 7. 14

4.17 Teste query 7. 14

4.18 Query: 8. 14

4.19 Teste query 8. 14

4.20 Query: 9. 15

4.21 Teste query 9. 15

4.22 Query: 10. 15

4.23 Teste query 10. 15

A.1 Predicado auxiliar: entregaValida. 18

A.2 Predicado auxiliar: veiculo. 18

A.3 Predicado auxiliar: classificacao. 18

A.4 Predicado auxiliar: data. 19

A.5 Predicado auxiliar: maior. 19

A.6 Predicado auxiliar: todasEntregasDup. 19

A.7 Predicado auxiliar: estafetasEntregaCliente. 19

A.8 Predicado auxiliar: retiraDup. 20

A.9 Predicado auxiliar: calculaValor. 20

A.10 Predicado auxiliar: custoTransporte. 20

A.11 Predicado auxiliar: encontraValores. 20

A.12 Predicado auxiliar: group. 20

A.13 Predicado auxiliar: calculaMedia. 21

A.14 Predicado auxiliar: 21

A.15 Predicado auxiliar: round. 21

A.16 Predicado auxiliar: veiculos. 21

A.17 Predicado auxiliar: calculaVDifEntrega. 21

A.18 Predicado auxiliar: calculaVentrega. 21

A.19 Predicado auxiliar: quantosIguais. 22

A.20 Predicado auxiliar: estafetas. 22

A.21 Predicado auxiliar: calculaEstafEntregas. 22

A.22 Predicado auxiliar: verificaPeriodo. 22

A.23 Predicado auxiliar: calculaHoras. 22

A.24 Predicado auxiliar: calculaHorasDia. 23

A.25 Predicado auxiliar: periodoEmHoras. 23

A.26 Predicado auxiliar: foiEntregue. 23

A.27 Predicado auxiliar: contaEncomendas. 23

A.28 Predicado auxiliar: calculaEentrega(. 23

A.29 Predicado auxiliar: pesoTransEstafeta. 23
A.30 Predicado auxiliar: agrupa. 24
A.31 Predicado auxiliar: atualizaPesos. 24
A.32 Predicado auxiliar: pertenceC. 24
A.33 Predicado auxiliar: somatorio. 24
A.34 Predicado auxiliar: concatenar. 24
A.35 Predicado auxiliar: checkData. 25
A.36 Predicado auxiliar: checkPeriodo. 25

Capítulo 2

Introdução

O exercício proposto teve como principal objetivo desenvolver um sistema de representação de conhecimento e raciocínio capaz de retratar o universo de discurso na área da logística de distribuição de encomendas. Para tal, e com recurso à programação em lógica PROLOG, foi criada uma base de conhecimento para a modelação a desenvolver. O sistema a modelar deve integrar um conjunto de funcionalidades, descritas ao longo deste relatório, que incitaram a criação de diversos predicados para a aplicação de *queries* sobre a base de conhecimento desenvolvida.

Capítulo 3

Preliminares

A resolução do exercício proposto, tal como foi previamente referido, recorre à programação em lógica PROLOG. Como tal, a representação do conhecimento foi efetuada com recurso a Lógica e a sua manipulação é elaborada através de mecanismos de Inferência. Como tal, a compreensão destas duas componentes é essencial para o acompanhamento do trabalho desenvolvido. Adicionalmente, a compreensão do mecanismo de inferência em PROLOG é imprescindível, pelo que a leitura de documentação é recomendada.

Capítulo 4

Descrição do Trabalho e Análise de Resultados

Ao longo desta secção será dada a conhecer a estratégia adotada pelo grupo para a resolução do exercício proposto. De modo a facilitar a compreensão e representação do universo em estudo, o trabalho foi dividido em três partes diferentes. Esta estruturação permite uma modelação mais clara do universo em estudo, seguindo uma perspetiva de desenvolvimento modular do código.

A primeira parte consiste no desenvolvimento da Base de Conhecimento, onde são dados a conhecer os predicados que integram definem o conhecimento preexistente acerca do universo em estudo. É sobre o conhecimento presente na Base de Conhecimento que o mecanismo de inferência usado trabalha para gerar novo conhecimento. Todo o conhecimento presente nesta componente representa conhecimento perfeito acerca do universo de discurso na área da logística de distribuição de encomendas.

Seguidamente, foi desenvolvida uma componente responsável pelo sistema de inferência. Este componente será responsável por executar as diversas funcionalidades propostas, i.e., deve ser capaz de responder a diversas *queries* propostas no enunciado do trabalho.

Por fim, optou-se por inserir todos os predicados auxiliares numa componente isolada. Esta segregação permite, em primeira parte, uma leitura e compreensão mais acessível das funcionalidades disponibilizadas. Adicionalmente, pretende que permita algum tipo de reutilização dos predicados, tentando que estes sejam o a mais genéricos possível.

4.1 Base de Conhecimento

Para a base de conhecimento foram considerados três predicados: *entrega*, *estafetaRanking* e *morada*.

O predicado *entrega* representa a informação associada à entrega de uma encomenda. Como tal, contém dados como o nome do estafeta que efetua a entrega, o veículo usado, o destinatário, as datas de encomenda e entrega, etc. Segue-se a caracterização completa do predicado:

entrega: estafeta, veiculo, distancia, tipoEncomenda, pesoEncomenda, prazo, velocidade, cliente, rua, classificacao, dataEncomenda, dataEntrega $\rightarrow V, F$

Adicionalmente, foi inserido o predicado *estafetaRanking* que associada a cada estafeta da Base de Conhecimento uma cotação – sendo esta cotação um número entre 0 e 5. Segue-se a caracterização completa do predicado:

estafetaRanking: estafeta, classificacao $\rightarrow V, F$

Por fim, foi integrado ainda o predicado *morada* que deescreve as informações relevantes acerca de cada utilizador. Segue-se a caracterização completa do predicado:

morada: cliente, rua, distancia $\rightarrow V, F$

Seguidamente, nas Figuras 4.1, 4.2 e 4.3, apresentam-se os factos integrados na Base de Conhecimento, referindo cada um dos predicados mencionados anteriormente.

```
%------------------------------ - - - - - - - - - - - - - - - - - -
% Extensão do predicado entrega: estafeta, veiculo, distancia, tipoEncomenda, pesoEnc, prazo, velocidade, cliente, rua, classificacao, dataEncomenda, dataEntrega -> {V, F}
entrega(manuel, bicicleta, 12, comida, 2, 1, 10, maria, 'avenida da liberdade, braga', 4, 11/10/2021/10, 11/10/2021/11).
entrega(manuel, bicicleta, 12, comida, 2, 1, 10, maria, 'avenida da liberdade, braga', 4, 11/10/2021/13, 11/10/2021/14).
entrega(manuel, bicicleta, 12, comida, 2, 1, 10, maria, 'avenida da liberdade, braga', 4, 11/10/2021/16, 11/10/2021/18).
entrega(manuel, bicicleta, 12, comida, 2, 1, 10, maria, 'avenida da liberdade, braga', 4, 11/10/2021/12, 11/10/2021/13).
entrega(jose, bicicleta, 15, comida, 4, 1, 10, ana, 'rua direita, barcelos', 5, 11/10/2021/12, 11/10/2021/13).
entrega(fabio, mota, 20, roupa, 12, 24, 35, filipa, 'rua de campelo, guimaraes', 5, 12/10/2021/13, 12/10/2021/19).
entrega(marco, carro, 23, novel, 80, 48, 25, cristina, 'travessa da igreja, famalicao', 3, 11/10/2021/17, 14/10/2021/10).
entrega(jose, bicicleta, 15, comida, 4, 1, 10, ana, 'rua direita, barcelos', 5, 11/10/2021/20, 11/10/2021/21).
entrega(manuel, bicicleta, 15, comida, 1, 1, 10, ana, 'rua direita, barcelos', 5, 11/10/2021/19, 11/10/2021/20).
entrega(fabio, bicicleta, 15, comida, 2, 1, 10, ana, 'rua direita, barcelos', 3, 12/10/2021/21, 12/10/2021/22).
entrega(manuel, bicicleta, 15, comida, 1, 1, 10, ana, 'rua direita, barcelos', 5, 11/10/2021/13, 11/10/2021/15).
entrega(manuel, mota, 12, roupa, 10, 18, 35, maria, 'avenida da liberdade, braga', 3, 12/10/2021/11, 13/10/2021/10).
entrega(manuel, mota, 23, comida, 3, 0.5, 35, cristina, 'travessa da igreja, famalicao', 4, 12/10/2021/14, 12/10/2021/15).
entrega(manuel, carro, 20, novel, 74, 30, 25, filipa, 'rua de campelo, guimaraes', 5, 13/10/2021/14, 16/10/2021/10).
entrega(manuel, carro, 23, comida, 3, 0.5, 25, cristina, 'travessa da igreja, famalicao', 4, 15/10/2021/18, 15/10/2021/19).
entrega(jose, carro, 15, roupa, 17, 4, 25, ana, 'rua direita, barcelos', 2, 15/10/2021/18, 17/10/2021/10).
entrega(jose, mota, 12, conida, 2, 1, 35, maria, 'avenida da liberdade, braga', 3, 14/10/2021/11, 14/10/2021/12).
entrega(jose, mota, 20, conida, 3, 0.5, 35, filipa, 'rua de campelo, guimaraes', 1, 12/10/2021/10, 12/10/2021/12).
entrega(jose, carro, 20, novel, 74, 30, 25, filipa, 'rua de campelo, guimaraes', 5, 13/10/2021/9, 16/10/2021/10).
entrega(jose, carro, 15, novel, 60, 5, 25, ana, 'rua direita, barcelos', 3, 14/10/2021/15, 15/10/2021/10).
entrega(fabio, carro, 15, novel, 45, 17, 25, ana, 'rua direita, barcelos', 2, 15/10/2021/15, 18/10/2021/10).
entrega(fabio, mota, 12, roupa, 9, 5, 35, maria, 'avenida da liberdade, braga', 3, 12/10/2021/16, 14/10/2021/10).
entrega(fabio, mota, 20, conida, 4, 2, 35, filipa, 'rua de campelo, guimaraes', 1, 12/10/2021/21, 12/10/2021/22).
entrega(fabio, carro, 23, novel, 53, 25, 25, cristina, 'travessa da igreja, famalicao', 2, 13/10/2021/4, 15/10/2021/10).
entrega(fabio, bicicleta, 15, comida, 2, 1, 10, ana, 'rua direita, barcelos', 3, 12/10/2021/13, 12/10/2021/14).
entrega(marco, bicicleta, 15, roupa, 3, 1, 10, ana, 'rua direita, barcelos', 2, 14/10/2021/9, 14/10/2021/18).
entrega(marco, mota, 12, conida, 4, 0.5, 35, maria, 'avenida da liberdade, braga', 4, 12/10/2021/10, 12/10/2021/12).
entrega(marco, mota, 23, conida, 3, 0.5, 35, cristina, 'travessa da igreja, famalicao', 5, 12/10/2021/11, 12/10/2021/12).
entrega(marco, carro, 20, novel, 74, 30, 25, filipa, 'rua de campelo, guimaraes', 5, 13/10/2021/11, 16/10/2021/10).
entrega(marco, carro, 23, novel, 89, 23, 25, cristina, 'travessa da igreja, famalicao', 5, 15/10/2021/14, 19/10/2021/10).
```

Figura 4.1: Base de Conhecimento: entrega.

```
%------------------------------------ - - - - - - - - - - - - - - - - - -
% Extensão do predicado estafetaRanking: estafeta, ranking -> {V, F}
estafetaRanking(jose, 4.3).
estafetaRanking(marco, 4.1).
estafetaRanking(fabio, 3.8).
estafetaRanking(manuel, 4.5).
```

Figura 4.2: Base de Conhecimento: estafetaRanking.

```
%------------------------------------ - - - - - - - - - - - - - - - - - -
% Extensão do predicado morada: cliente, rua, distancia -> {V, F}
morada(maria, 'avenida da liberdade, braga', 12).
morada(ana, 'rua direita, barcelos', 15).
morada(filipa, 'rua de campelo, guimaraes', 20).
morada(cristina, 'travessa da igreja, famalicao', 23).
```

Figura 4.3: Base de Conhecimento: morada.

4.2 Queries

A seguinte secção procura dar a conhecer as funcionalidades desenvolvidas no âmbito do exercício. As presentes funcionalidades possibilitam questionar a Base de Conhecimento, com base nos seu factos, e geram posterior conhecimento[1].

A primeira query efetuada à Base de Conhecimento do universo de discurso em estudo pretende determinar qual o estafeta que utilizou mais vezes o meio de transporte mais ecológico. Para responder a esta questão, tal como pode ser verificado através da Figura 4.4, inicialmente foi efetuada uma listagem de todas as encomendas válidas entregues segundo um determinado meio de transporte, através de um predicado auxiliar. Uma vez obtida esta listagem, foi escolhido, posteriormente, o estafeta que mais entregas efetuou.

```
%------------------------------ - - - - - - - - - - - - - - - - -
% 1) Identificar o estafeta que utilizou mais vezes um meio de transporte mais ecológico
% Extensão do predicado estafetaMaisVezesTransp: Veiculo, Estafeta -> {V,F}
estafetaMaisVezesTransp(V, R) :-
        findall(E, entregaValida(E, V, KM, T, P, Pr, Vel, C, R, Cla, Denc, Dent), [H|T]),
        maior(H, 1, [H|T], R).
```

Figura 4.4: Query 1.

Para determinar quem usou o meio mais ecológico basta especificar no predicado o meio de transporte "bicicleta". Deste modo, a funcionalidade torna-se mais genérica, podendo sempre quem efetuou mais entregas em função de cada tipo de transporte. A Figura 4.5 demonstra um teste efetuado, para determinar quem foi o estafeta que mais entregas efetuou de bicicleta.

```
?- estafetaMaisVezesTransp(bicicleta,R).
R = manuel.
```

Figura 4.5: Teste query 1.

A segunda funcionalidade permite efetuar a identificação dos estafetas que entregaram uma determinada encomenda a um certo cliente. Para gerar esta funcionalidade, tal como pode ser observado na Figura 4.6, foi usado um predicado auxiliar que seleciona da Base de Conhecimento todas as encomendas entregues a um dado cliente. Após a obtenção desta lista, foram removidos os duplicados, originando o resultado final.

```
%------------------------------ - - - - - - - - - - - - - - - - -
% 2) Identificar que estafetas entregaram determinada(s) encomenda(s) a um determinado cliente
% Extensão do predicado estafetasEntregasCliente: Cliente, Lista encomendas, Lista de estafetas -> {V,F}
todasEntregas(C, T, R) :-
        todasEntregasDup(C, T, D),
        retiraDup(D, [], R), !.
```

Figura 4.6: Query 2.

A funcionalidade foi testada e o seu teste pode ser consultado na Figura 4.7.

[1] cf. Anexo A para ver todos os predicados auxiliares usados nas diversas funcionalidades.

```
?- todasEntregas(maria,[comida],R).
R = [manuel, jose, marco].
```

Figura 4.7: Teste query 2.

A seguinte funcionalidade proposta, permite determinar quais são os clientes que receberam entregas de um determinado estafeta. Para tal, foram filtradas todas as encomendas válidas entregues por um dado estafeta, sendo gerada uma lista com os nomes de clientes associados a essas encomendas. Uma vez que a lista admite nomes duplicados, foram posteriormente removidos os nomes duplicados, tal como demonstra a Figura 4.8.

```
%-------------------------------------------------------------
% 3) Identificar os clientes servidos por um determinado estafeta
% Extensão do predicado clientesServidosEstafeta: Estafeta, Lista Clientes -> {V,F}
clientesServidosEstafeta(E, Clientes) :-
        findall(C, entregaValida(E, V, KM, T, P, Pr, Vel, C, R, Cla, Denc, Dent), L),
        retiraDup(L, [], Clientes), !.
```

Figura 4.8: Query 3.

A Figura 4.9 demonstra o teste efetuado para a funcionalidade acima descrita.

```
?- clientesServidosEstafeta(manuel,R).
R = [filipa, cristina, ana, maria].
```

Figura 4.9: Teste query 3.

A quarta funcionalidade desenvolvida possibilita determinar o valor total de faturação que a empresa *Green Distribution* obteve num determinado dia. O cálculo deste valor foi decomposto em diversos passos, observáveis na Figura 4.10. Inicialmente, foram filtradas todas as encomendas válidas efetuadas nesse dia, registando num triplo o tipo de veículo usado, o peso da encomenda e os quilómetros percorridos. Uma vez obtida uma lista com estes triplos foi calculado o valor total das encomendas – o valor de cada encomenda é descrito em função dos parâmetros do triplo, sendo estes usados no predicado auxiliar para o seu cálculo.

```
%-------------------------------------------------------------
% 4) calcular o valor faturado pela Green Distribution num determinado dia
% Extensão do predicado valorFaturado: data, resultado -> {V,F}
valorFaturado(D/M/A/_, R) :-
        findall((V, P, KM), encontraValores(D/M/A/_, (V, P, KM)), L),
        calculaValor(L, R).
```

Figura 4.10: Query 4.

O teste da funcionalidade previamente descrita encontra-se na Figura 4.11.

```
?- valorFaturado(11/10/2021/_,R).
R = 57.6.
```

Figura 4.11: Teste query 4.

A query 5 elaborada consiste em determinar qual a zona com maior volume de encomendas. Para tal, considerou-se que o volume de entregas correspondia ao número total de entregas nessa zona. Assim sendo, e tal como demonstra a Figura 4.12, efetuou-se inicialmente a filtragem de todas as zonas onde foram entregues encomedas. Uma vez obtida esta lista, e com recurso a um predicado auxiliar, foram criados tuplos, caracterizados pela zona e o número de repetições dessa zona na lista – o que permite determinar o número total de encomedas por zona. Após a ordenação, crescente, das zonas em função do número de encomendas, foi selecionado o primeiro elemento da lista, para devolver a zona com maior volume de entregas.

```
%------------------------------ - - - - - - - - - - - - - - - - - -
% 5) Identificar quais as zonas (e.g., rua ou freguesia) com maior volume de entregas por parte da Green Distribution
% Extensão do predicado volumeZona: resultado -> {V,F}
volumeZona(R) :-
        findall(Zone,entregaValida(_, _, _, _, _, _, _, _, Zone, _, _, _), Zones), % lista com todas as zonas
        group(Zones,Aux), % agrupa o número de zonas iguais indicando a zona e o número de repetições
        sort(2,@>=,Aux, Aux1), % ordenar a lista em função do número de encomendas
        nth0(0,Aux1,R,Tail). % selecionar a cabeça da lista
```

Figura 4.12: Query: 5.

A funcionalidade apresentada foi testada, sendo o resultado do seu teste apresentado na Figura 4.13.

```
?- volumeZona(R).
R =  ('rua direita, barcelos', 10).
```

Figura 4.13: Teste: query 5.

A funcionalidade seguinte permite determinar a classificação média de satisfação dos clientes face a um estafeta. Inicialmente, efetuou-se uma listagem de todas as entregas válidas realizadas por um dado estafeta e, como pode ver visto na Figura 4.14, essa listagem foi usada posteriormente, por um predicado auxiliar, para determinar a médica de classificação.

```
%------------------------------ - - - - - - - - - - - - - - - - - -
% 6) Calcular a classificação média de satisfação de cliente para um determinado estafeta
% Extensão do predicado classificacaoMedia: estafeta, resultado -> {V,F}
classificacaoMedia(E, R) :-
        findall(Class, entregaValida(E, V, KM, T, P, Pr, Vel, C, Zone, Class, Denc, Dent), L),
        calculaMedia(L, R).
```

Figura 4.14: Query: 6.

A Figura 4.15 apresenta o teste efetuado para a funcionalidade apresentada.

```
?- classificacaoMedia(jose,R).
R = 3.4.
```

Figura 4.15: Teste query 6.

A sétima funcionalidade apresentada permite, dado um intervalo de tempo, identificar o número total de entregas realizadas pelos vários meios de transporte. A Figura 4.16 demonstra a abordagem adotada pelo grupo para proporcionar esta funcionalidade e, tal como pode ser verificado, o processo foi efetuado com recurso a diversos predicados auxiliares. Inicialmente, são efetuadas

verificações acerca das datas fornecidas: uma vez validadas, verifica-se se o intervalo de tempo fornecido é, também, válido. Seguidamente, e obtida uma lista de todos os veículos que se encontram na Base de Conhecimento. Uma vez obtidos estes dados, é determinado, por fim, o número de encomendas que os efetuadas.

```
%------------------------------ - - - - - - - - - - - - - - - - - -
% 7) Identificar o número total de entregas pelos diferentes meios de transporte, num determinado intervalo de tempo
% Extensão do predicado totalVeiculoEntrega: data1, data2, resultado -> {V,F}
totalVeiculoEntrega(Data1,Data2,Nentregas):-
        data(Data1),
        data(Data2),
        checkPeriodo(Data1,Data2),
        veiculos(Veiculos),
        calculaVDifEntrega(Data1,Data2,[],Veiculos, Nentregas), !.
```

Figura 4.16: Query: 7.

O teste da funcionalidade é apresentado abaixo na Figura 4.17.

```
?- totalVeiculoEntrega(11/10/2021/10,16/10/2021/10,R).
R = [(bicicleta, 11),  (mota, 9),  (carro, 7)].
```

Figura 4.17: Teste query 7.

A funcionalidade seguinte permite, dado um intervalo de tempo, identificar o número total de entregas realizadas pelos vários estafetas. Para desenvolver esta funcionalidade segiu-se um raciocínio semelhante ao modo de construção demonstrado na funcionalidade anterior. Então, deste modo, após as verificações necessárias associadas ao período de tempo fornecido, foram selecionados todos estafetas da Base de Conhecimento para, posteriormente, através de um predicado auxiliar, determinar o número total de encomendas efetuadas – tal como pode ser visto na Figura 4.18.

```
%------------------------------ - - - - - - - - - - - - - - - - - -
% 8) Identificar o número total de entregas pelos estafetas, num determinado intervalo de tempo
% Extensão do predicado totalEntregasEstafetas: data1, data2, resultado -> {V,F}
totalEntregasEstafetas(Data1,Data2,Nentregas):-
        data(Data1),
        data(Data2),
        checkPeriodo(Data1,Data2),
        estafetas(Estafetas),
        calculaEstafEntregas(Data1,Data2,[],Estafetas, Nentregas),
        !.
```

Figura 4.18: Query: 8.

A Figura 4.19 apresenta o teste efetuado para a funcionalidade descrita anteriormente.

```
?- totalEntregasEstafetas(11/10/2021/10,16/10/2021/10,R).
R = [(manuel, 10),  (jose, 6),  (fabio, 6),  (marco, 5)].
```

Figura 4.19: Teste query 8.

Adicionalmente, foi inserida uma funcionalidade que permite calcular o número de encomendas que foram entregues e não entregues pela empresa, num período de tempo. A Figura 4.20 apresenta a especificação da funcionalidade. Para determinar se uma encomenda é, ou não entregue, foi usada

informação presente no predicado *entrega*. Uma encomenda é considerada entregue se a diferença entre a data de entrega e a data de encomenda não excede o prazo definido para a encomenda. Assim sendo, as datas fornecidas são convertidas para horas e a diferença entre ambas é comparada com o prazo definido – estas operações são efetuadas com recurso a predicados auxiliares. Uma vez determinado se o prazo definido é, ou não, superior ao tempo de entrega, os tuplos da lista resultante são agregados, somando os seus valores.

```
%------------------------------ - - - - - - - - - - - - - - - - - - - -
% 9) Calcular o número de encomendas entregues e não entregues pela Green Distribution, num determinado período de tempo
% Extensão do predicado numEncomendas: data1, data2, encomendas, resultado  -> {V,F}
numEncomendas(D1, D2, Ent, NEnt) :-
        findall((Tent, TNent),(verificaPeriodo(D1, D2, (P, DEnc, DEnt)),periodoEmHoras((DEnc,DEnt),P2) ,foiEntregue((P,P2),Tent,TNent)), L),
        contaEncomendas(L,0,0,Ent, NEnt).
```

Figura 4.20: Query: 9.

O teste da funcionalidade descrita acima encontra-se na Figura 4.21.

```
[?- numEncomendas(11/10/2021/10,16/10/2021/10,Ent,NEnt).
Ent = 11,
NEnt = 16.
```

Figura 4.21: Teste query 9.

Por fim, foi adicionada a funcionalidade de calcular o peso total calculado por estafeta num determinado dia. Tal como mostra a Figura 4.22, com recurso a predicados auxiliares, iniciou-se por determinar quais os estafetas efetuaram entregas na data fornecida, e qual o peso individual de cada entrega. Posteriormente, a lista resultante da operação anterior foi agrupada em função do nome dos estafetas, somando os pesos correspondentes a cada encomenda.

```
%------------------------------ - - - - - - - - - - - - - - - - - - - -
% 10) Calcular o peso total transportado por estafeta num determinado dia
% Extensão do predicado pesoTransEstafetaDia: data, resultado -> {V,F}
pesoTransEstafetaDia(D/M/A/_, R) :-
        findall((E, P), pesoTransEstafeta(D/M/A/_, (E, P)), L),
        agrupa(L, [], R).
```

Figura 4.22: Query: 10.

A Figura 4.23 apresenta o teste efetuado.

```
[?- pesoTransEstafetaDia(11/10/2021/10,R).
R = [(jose, 8),  (manuel, 10)].
```

Figura 4.23: Teste query 10.

Capítulo 5

Conclusões e Sugestões

Ao longo do presente relatório foram descritos os diversos passos adotados pelo grupo para o desenvolvimento de um sistema de representação de conhecimento e raciocínio, capaz de retratar o universo de discurso na área da logística de distribuição de encomendas. Para tal, foi criada uma Base de Conhecimento para o efeito, assim como foram desenvolvidas um conjunto de funcionalidades, explicadas ao longo do relatório.

O trabalho aqui apresentado corresponde aos requisitos pedidos, disponibilizando uma série de funcionalidades que trabalha sobre a Base de Conhecimento criada. No entanto, o sistema modelado encontra-se apto para ser estendido a casos mais complexos. Para tal, por um lado, podem ser incorporados predicados adicionais que permitam efetuar outras interrogações ao sistema. Adicionalmente, seria também interessante permitir que, através do uso de invariantes, o sistema fosse capaz de evoluir e involuir, ou seja, integrando estes fatores o sistema seria capaz de adicionar e remover informação sem comprometer a consistência da Base de Conhecimento.

Capítulo 6

Bibliografia

[1] ANALIDE, Cesar, NOVAIS, Paulo
"Sugestões para a Redacção de Relatórios Técnicos"
Relatório Técnico, Departamento de Informática, Universidade do Minho, Portugal, 2011

[2] BRATKO, Ivan
"PROLOG: Programming for Artificial Intelligence"
United States, Pearson Education (US), 2000

Apêndice A

Predicados auxiliares

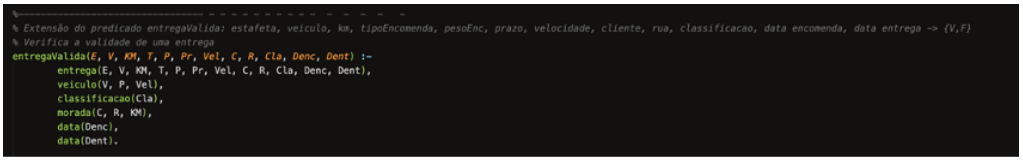


Figura A.1: Predicado auxiliar: entregaValida.

Figura A.2: Predicado auxiliar: veiculo.

Figura A.3: Predicado auxiliar: classificacao.

```
%---------------------------------- - - - - - - - - - - - - - - - -
% Extensão do predicado data: dia, mes, ano, hora -> {V,F}
% Verifica se uma data é válida
data(D/2/A/H) :-
        A >= 0, !,
        A mod 4 =:= 0, !,
        D >= 1, !,
        D =< 29, !,
        member(H, [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23]), !.
data(D/M/A/H) :-
        A >= 0, !,
        member(M, [1,3,5,7,8,10,12]), !,
        D >= 1, !,
        D =< 31, !,
        member(H, [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23]), !.
data(D/M/A/H) :-
        A >= 0, !,
        member(M, [4,6,9,11]), !,
        D >= 1, !,
        D =< 30, !,
        member(H, [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23]), !.
data(D/2/A/H) :-
        A >= 0, !,
        A mod 4 =\= 0, !,
        D >= 1, !,
        D =< 28, !,
        member(H, [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23]), !.
```

Figura A.4: Predicado auxiliar: data.

```
%---------------------------------- - - - - - - - - - - - - - - - -
% Extensão do predicado maior: nome, numeroRepetições, lista, resultado -> {V,F}
% Predicado para verificar qual o nome do estafeta aparece mais vezes numa lista
% Predicado auxiliar usado na query 1
maior(Nome, N, [], Nome).
maior(Nome, N, [H|T], R) :-
        quantosIguais([H|T], N1),
        N >= N1,
                maior(Nome, N, T, R), !.
maior(H, N1, [H|T], R) :-
        quantosIguais([H|T], N1),
                N < N1,
                maior(H, N1, T, R), !.
```

Figura A.5: Predicado auxiliar: maior.

```
%---------------------------------- - - - - - - - - - - - - - - - -
% Extensão do predicado todasEntregasDup: cliente, lista, resultado -> {V,F}
% Concatena todas as listas que do predicado estafetasEntregaCliente
% Predicado auxiliar usado na query 2
todasEntregasDup(C, [], []).
todasEntregasDup(C, [T|Tail], R) :-
        estafetasEntregaCliente(C, T, Temp),
        concatenar(Temp, Temp1, R),
        todasEntregasDup(C, Tail, Temp1).
```

Figura A.6: Predicado auxiliar: todasEntregasDup.

```
%---------------------------------- - - - - - - - - - - - - - - - -
% Extensão do predicado estafetasEntregaCliente: cliente, tipoEncomenda, estafeta -> {V,F}
% Devolve uma lista com os estafetas que entregaram um tipo de encomenda a um determinado cliente
estafetasEntregaCliente(C, T, R) :-
        findall(E, estafetasEntregaClienteValida(C, T, E), L),
        retiraDup(L, [], R).
```

Figura A.7: Predicado auxiliar: estafetasEntregaCliente.

Figura A.8: Predicado auxiliar: retiraDup.

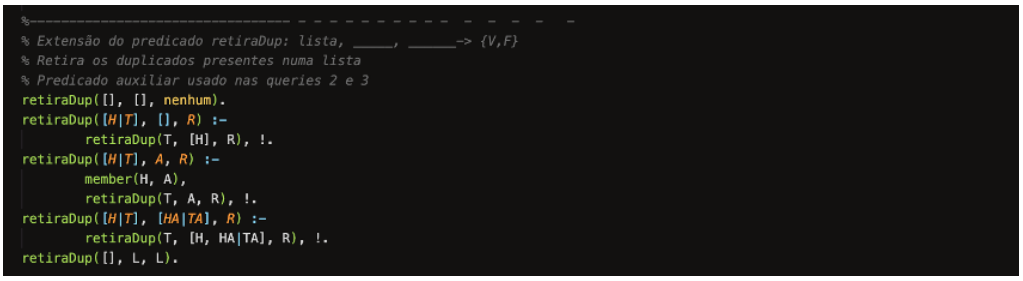


Figura A.9: Predicado auxiliar: calculaValor.

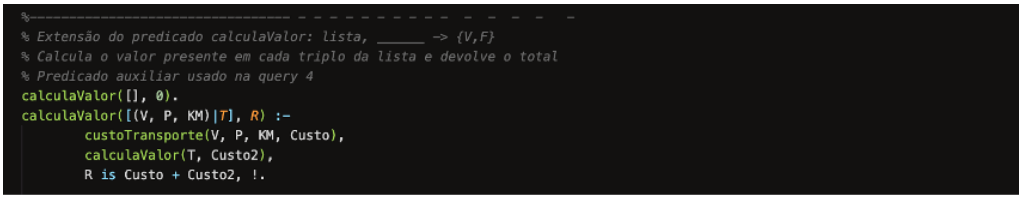


Figura A.10: Predicado auxiliar: custoTransporte.

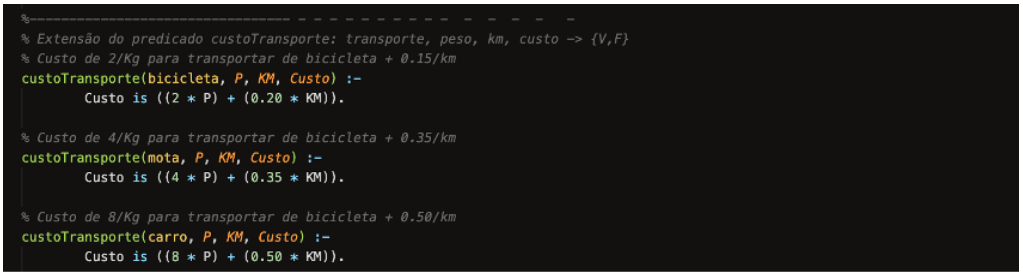


Figura A.11: Predicado auxiliar: encontraValores.

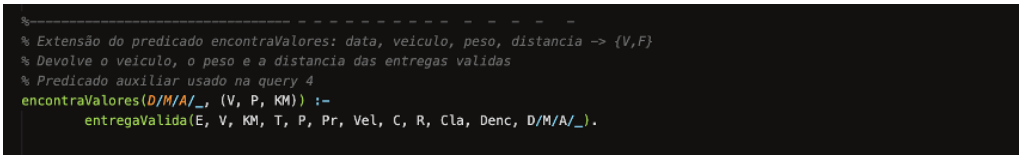


Figura A.12: Predicado auxiliar: group.

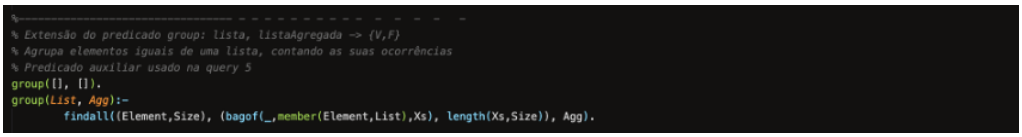


Figura A.13: Predicado auxiliar: calculaMedia.

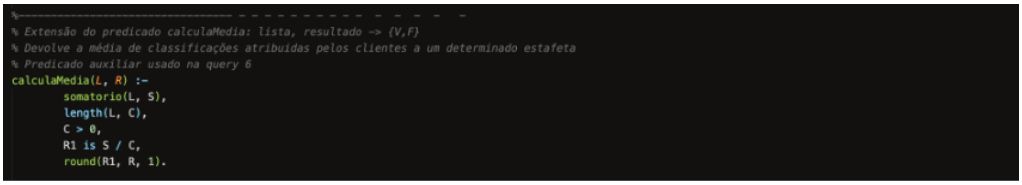


Figura A.14: Predicado auxiliar: .

Figura A.15: Predicado auxiliar: round.

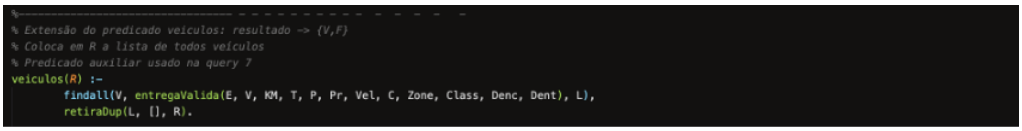


Figura A.16: Predicado auxiliar: veiculos.

Figura A.17: Predicado auxiliar: calculaVDifEntrega.

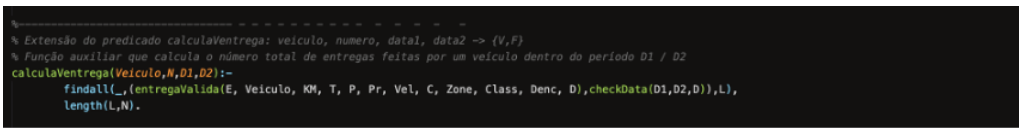


Figura A.18: Predicado auxiliar: calculaVentrega.

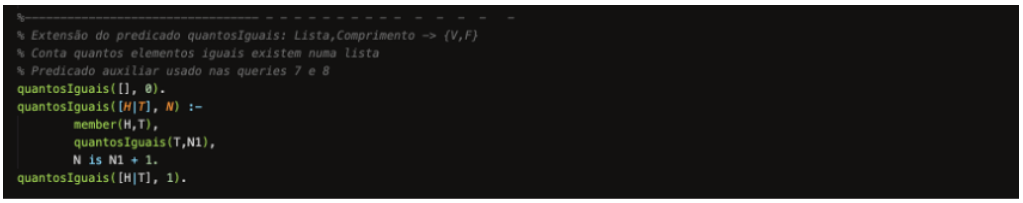


Figura A.19: Predicado auxiliar: quantosIguais.

Figura A.20: Predicado auxiliar: estafetas.

Figura A.21: Predicado auxiliar: calculaEstafEntregas.

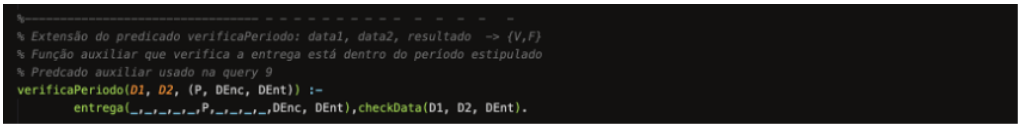


Figura A.22: Predicado auxiliar: verificaPeriodo.

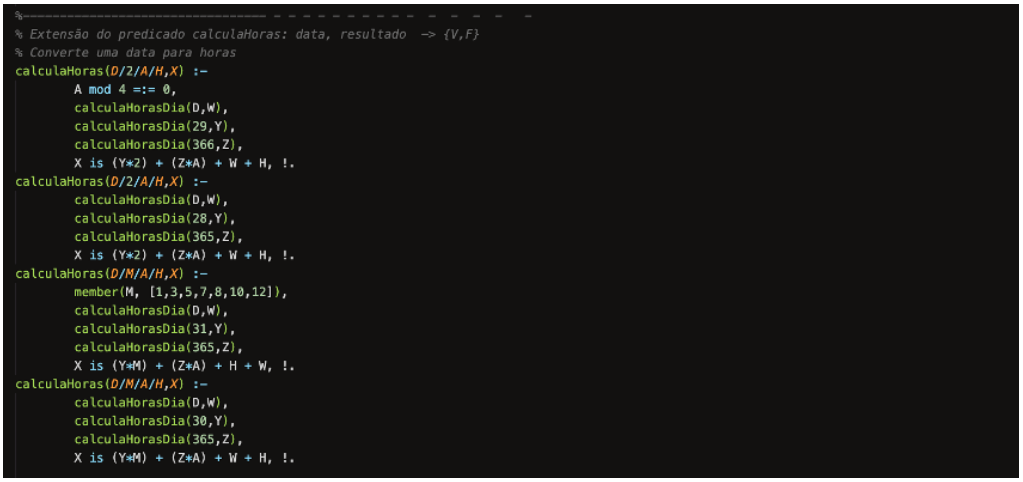


Figura A.23: Predicado auxiliar: calculaHoras.

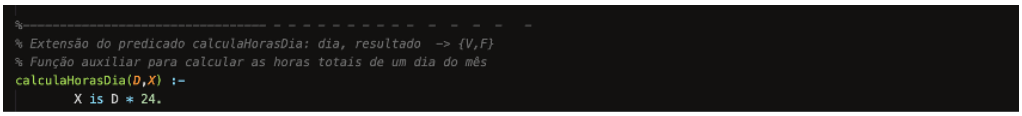


Figura A.24: Predicado auxiliar: calculaHorasDia.

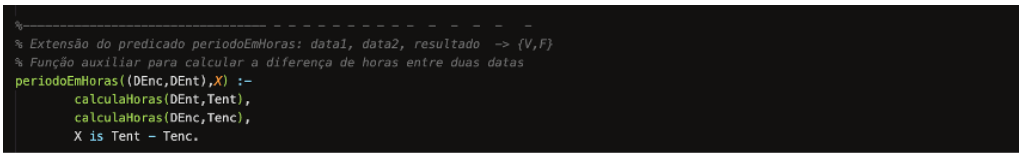


Figura A.25: Predicado auxiliar: periodoEmHoras.

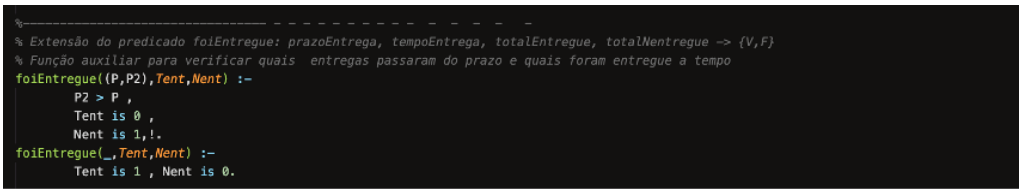


Figura A.26: Predicado auxiliar: foiEntregue.

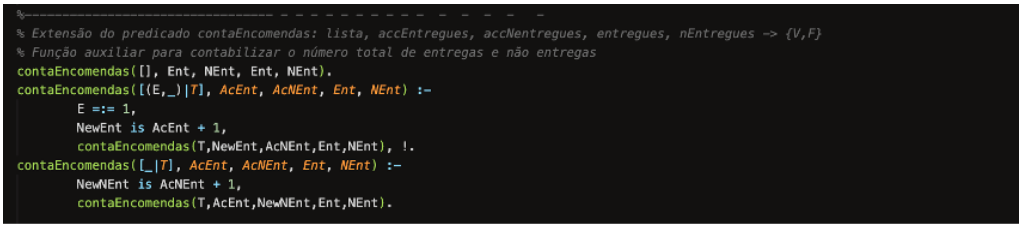


Figura A.27: Predicado auxiliar: contaEncomendas.

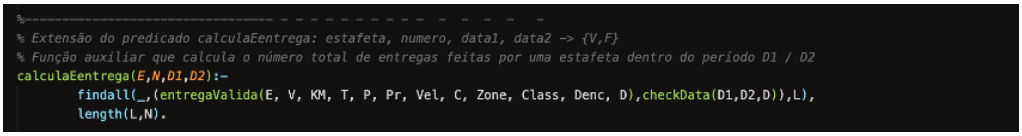


Figura A.28: Predicado auxiliar: calculaEentrega(.

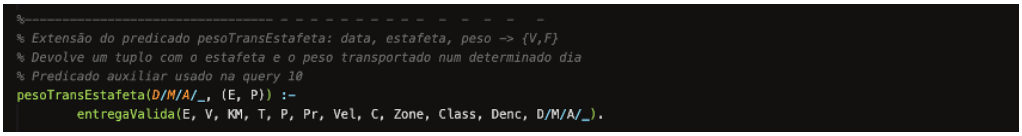
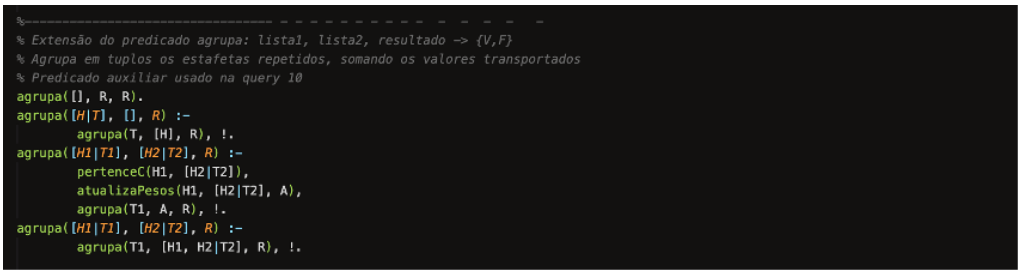
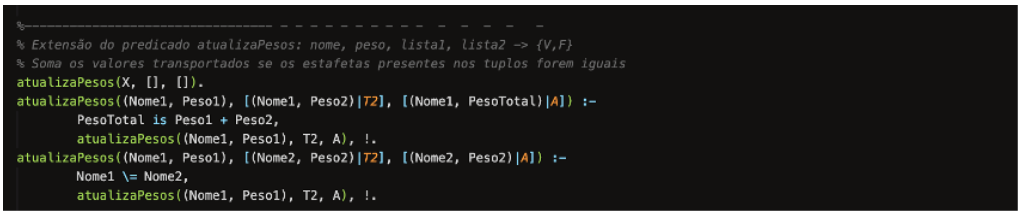


Figura A.29: Predicado auxiliar: pesoTransEstafeta.



```
%-------------------------------- - - - - - - - - - - - - - - - - - -
% Extensão do predicado agrupa: lista1, lista2, resultado -> {V,F}
% Agrupa em tuplos os estafetas repetidos, somando os valores transportados
% Predicado auxiliar usado na query 10
agrupa([], R, R).
agrupa([H|T], [], R) :-
        agrupa(T, [H], R), !.
agrupa([H1|T1], [H2|T2], R) :-
        pertenceC(H1, [H2|T2]),
        atualizaPesos(H1, [H2|T2], A),
        agrupa(T1, A, R), !.
agrupa([H1|T1], [H2|T2], R) :-
        agrupa(T1, [H1, H2|T2], R), !.
```

Figura A.30: Predicado auxiliar: agrupa.

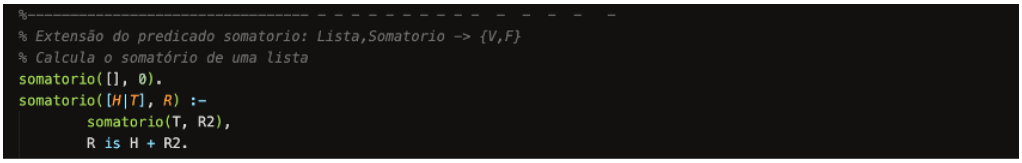


```
%-------------------------------- - - - - - - - - - - - - - - - - - -
% Extensão do predicado atualizaPesos: nome, peso, lista1, lista2 -> {V,F}
% Soma os valores transportados se os estafetas presentes nos tuplos forem iguais
atualizaPesos(X, [], []).
atualizaPesos((Nome1, Peso1), [(Nome1, Peso2)|T2], [(Nome1, PesoTotal)|A]) :-
        PesoTotal is Peso1 + Peso2,
        atualizaPesos((Nome1, Peso1), T2, A), !.
atualizaPesos((Nome1, Peso1), [(Nome2, Peso2)|T2], [(Nome2, Peso2)|A]) :-
        Nome1 \= Nome2,
        atualizaPesos((Nome1, Peso1), T2, A), !.
```

Figura A.31: Predicado auxiliar: atualizaPesos.

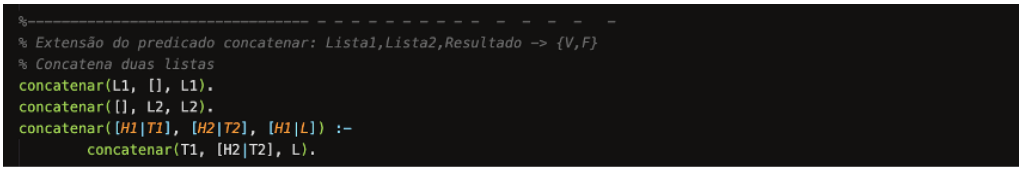
```
%-------------------------------- - - - - - - - - - - - - - - - - - -
% Extensão do predicado pertenceC: tupo, lista -> {V,F}
%
pertenceC( (X, P),[(X, N)|L] ).
pertenceC( (X, P),[(Y, N)|L] ) :-
        X \= Y,
        pertenceC( (X, P),L ).
```

Figura A.32: Predicado auxiliar: pertenceC.



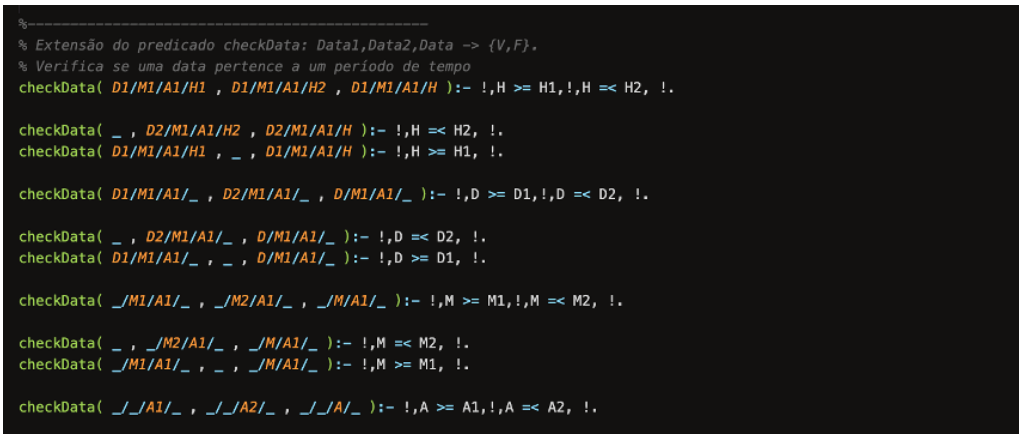
```
%-------------------------------- - - - - - - - - - - - - - - - - - -
% Extensão do predicado somatorio: Lista,Somatorio -> {V,F}
% Calcula o somatório de uma lista
somatorio([], 0).
somatorio([H|T], R) :-
        somatorio(T, R2),
        R is H + R2.
```

Figura A.33: Predicado auxiliar: somatorio.



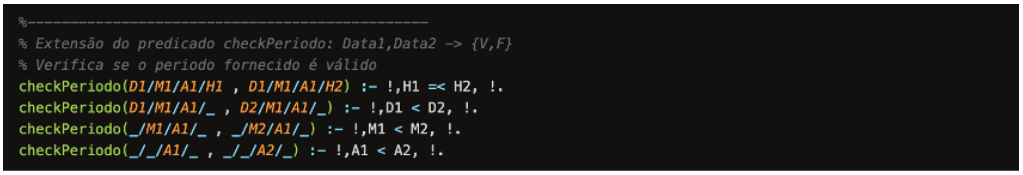
```
%-------------------------------- - - - - - - - - - - - - - - - - - -
% Extensão do predicado concatenar: Lista1,Lista2,Resultado -> {V,F}
% Concatena duas listas
concatenar(L1, [], L1).
concatenar([], L2, L2).
concatenar([H1|T1], [H2|T2], [H1|L]) :-
        concatenar(T1, [H2|T2], L).
```

Figura A.34: Predicado auxiliar: concatenar.



```
%-----------------------------------------------------
% Extensão do predicado checkData: Data1,Data2,Data -> {V,F}.
% Verifica se uma data pertence a um período de tempo
checkData( D1/M1/A1/H1 , D1/M1/A1/H2 , D1/M1/A1/H ):- !,H >= H1,!,H =< H2, !.

checkData( _ , D2/M1/A1/H2 , D2/M1/A1/H ):- !,H =< H2, !.
checkData( D1/M1/A1/H1 , _ , D1/M1/A1/H ):- !,H >= H1, !.

checkData( D1/M1/A1/_ , D2/M1/A1/_ , D/M1/A1/_ ):- !,D >= D1,!,D =< D2, !.

checkData( _ , D2/M1/A1/_ , D/M1/A1/_ ):- !,D =< D2, !.
checkData( D1/M1/A1/_ , _ , D/M1/A1/_ ):- !,D >= D1, !.

checkData( _/M1/A1/_ , _/M2/A1/_ , _/M/A1/_ ):- !,M >= M1,!,M =< M2, !.

checkData( _ , _/M2/A1/_ , _/M/A1/_ ):- !,M =< M2, !.
checkData( _/M1/A1/_ , _ , _/M/A1/_ ):- !,M >= M1, !.

checkData( _/_/A1/_ , _/_/A2/_ , _/_/A/_ ):- !,A >= A1,!,A =< A2, !.
```

Figura A.35: Predicado auxiliar: checkData.



```
%-----------------------------------------------
% Extensão do predicado checkPeriodo: Data1,Data2 -> {V,F}
% Verifica se o periodo fornecido é válido
checkPeriodo(D1/M1/A1/H1 , D1/M1/A1/H2) :- !,H1 =< H2, !.
checkPeriodo(D1/M1/A1/_ , D2/M1/A1/_) :- !,D1 < D2, !.
checkPeriodo(_/M1/A1/_ , _/M2/A1/_) :- !,M1 < M2, !.
checkPeriodo(_/_/A1/_ , _/_/A2/_) :- !,A1 < A2, !.
```

Figura A.36: Predicado auxiliar: checkPeriodo.