Relatório 1ª Fase - Grupo 10

Inteligência Artificial

LEI - UM

2021/2022

Francisco Novo A89567

João Silva A89293

João Vieira A93170

Tiago Ribeiro A93203

Universidade do Minho
Escola de Engenharia

Índice

Introdução 1

Predicados Base 2

Predicados Auxiliares 5

Funcionalidades 7

Conclusão 16

Introdução

Nos tempos que correm a ecologia é um tópico bastante importante na nossa sociedade. Por esse motivo, a empresa *Green Distribution* tem como objetivo privilegiar sempre o meio de entrega mais ecológico. A empresa tem ao seu dispor diversos meios de transporte tais como bicicletas, motos e carros, que podem ser classificados pela vertente ecológica.

Com base na caracterização enunciada da empresa, o objetivo do trabalho consiste no desenvolvimento de um sistema capaz de demonstrar as funcionalidades subjacentes à utilização da linguagem de programação em lógica PROLOG, no âmbito da representação de conhecimento e construção de mecanismos de raciocínio para a resolução do problema apresentado.

Este deverá ser capaz de caracterizar um universo de discurso na área da logística de distribuição de encomendas, entre outros objetos, adequado para escrever, ler e/ou interpretar conhecimentos relativos à empresa.

Predicados Base

Para a realização do trabalho recorreu-se a uma base de dados capaz de executar todas as tarefas pedidas.

A parte mais importante desta etapa foi a separação do predicado 'encomenda' e do predicado 'entrega', sendo que o segundo predicado realiza a entrega do primeiro, ajudando assim na resolução de algumas funcionalidades necessárias.

Assim, foram criados os seguintes predicados:

Estafeta: Constituído por : *Id_estafeta, Nome, Idade*.

```
%Estafeta(id_estafeta,nome,idade)

estafeta(01 ,joao_vieira ,20).
estafeta(02 ,francisco_novo ,21).
estafeta(03 ,tiago_ribeiro ,20).
estafeta(05 ,diana_pinto ,22).
estafeta(06 ,henrique_lopes ,31).
estafeta(07 ,paulo_oliveira ,18).
estafeta(08 ,francisco_izquierdo ,22).
estafeta(09 ,duarte_lucas ,28).
estafeta(10 ,pedro_magalhaes ,29).
estafeta(11 ,carlos_dias ,22).
estafeta(12 ,daniel_novo ,19).
estafeta(13 ,joao_goncalves ,25).
estafeta(14 ,beatriz_leite ,24).
estafeta(15 ,catia_pereira ,21).
estafeta(16 ,joao_torres ,61).
estafeta(17 ,maria_aurora, 18).
estafeta(18 ,vitorino_donald, 35).
```

Cliente: Constituído por : *Id_Cliente, Nome, Idade, Nome_Rua*.

```
%Cliente(id_cliente,nome,idade,Nome_Rua).

cliente(01 ,paulo_rodrigues ,20 ,jose_maria_ottoni).
cliente(02 ,carolina_sousa ,26 ,avenida_5_de_outubro).
cliente(03 ,hugo_machado ,32 ,rua_central).
cliente(04 ,clara_teixeira ,23 ,santa_luzia).
cliente(05 ,joana_rua ,43 ,sao_jorge).
cliente(06 ,flavia_araujo,22,duque_de_caxias).
cliente(07 ,joana_rua ,23 ,projetada).
cliente(08 ,ricardo_guimares ,26 ,rui_barbosa).
cliente(09 ,margarida_santos ,31 ,santa_catarina).
cliente(10 ,joao_gabriel ,33 ,boa_vista).
cliente(11 ,joel_braga ,26 ,sao_luis).
cliente(12 ,joao_tomas ,32 ,vinte_e_quatro).
cliente(13 ,leonardo_freitas ,21 ,sete_ceus).
cliente(14 ,ricardo_ferreira ,18 ,beco_belo).
cliente(15 ,helder_gomes ,26 ,amorosa).
cliente(16 ,david_goncalves ,39 ,travessa_do_rio).
cliente(17 ,goncalo_rodrigues , 26 , santa_maria).
cliente(18 ,marta_pinto ,29 ,lamelas).
cliente(19 ,joao_ramos , 31 ,bairro_feliz).
cliente(20 ,vasco_moreno ,32 ,augusta).
cliente(21 ,beatriz_morais ,38 ,calcada_santana).
cliente(22 ,andre_carreiras ,21 ,avenida_paz).
cliente(23 ,nuno_valente ,28,rua_barros).
cliente(24 ,bruno_mota ,24,caranda).
```

Encomenda: Constituído por : *Id_Encomenda, Id_cliente, Id_estafeta, Prazo_Pedido, Peso, Volume.*

```
% encomenda(id_encomenda,id_cliente,id_estafeta,prazo_pedido(horas),peso(kg),volume(m3))

encomenda(01 ,01 ,01 ,03 ,04 ,20).
encomenda(02 ,02 ,02 ,01 ,04 ,12).
encomenda(03 ,03 ,03 ,08 ,75 ,50).
encomenda(04 ,04 ,01 ,03 ,25 ,30).
encomenda(05 ,05 ,04 ,10 ,16 ,15).
encomenda(06 ,06 ,01 ,07 ,02 ,06).
encomenda(07 ,18 ,02 ,06 ,03 ,04).
encomenda(08 ,08 ,01 ,04 ,35 ,20).
encomenda(09 ,09 ,02 ,08 ,05 ,12).
encomenda(10 ,10 ,03 ,09 ,75 ,50).
encomenda(11 ,11 ,01 ,06 ,25 ,30).
encomenda(12 ,12 ,04 ,03 ,16 ,15).
encomenda(13 ,13 ,01 ,10 ,12 ,06).
encomenda(14 ,14 ,02 ,11 ,03 ,04).
encomenda(15 ,15 ,01 ,12 ,04 ,20).
encomenda(16 ,16 ,02 ,13 ,14, 12).
encomenda(17 ,17 ,03 ,06 ,75 ,50).
encomenda(18 ,07 ,01 ,05 ,25 ,30).
encomenda(19 ,19 ,04 ,04 ,16 ,15).
encomenda(20 ,20 ,01 ,06 ,12 ,06).
encomenda(21 ,21 ,02 ,08 ,03 ,04).
encomenda(22 ,22 ,01 ,09 ,24 ,20).
encomenda(23 ,23 ,02 ,01 ,18 ,12).
encomenda(24 ,11 ,03 ,07 ,75 ,50).
encomenda(25 ,25 ,01 ,06 ,25 ,30).
encomenda(26 ,26 ,04 ,09 ,16 ,15).
encomenda(27 ,01 ,01 ,08 ,12 ,06).
encomenda(28 ,02 ,02 ,02 ,03 ,04).
encomenda(29 ,23 ,01 ,05 ,04 ,20).
encomenda(30 ,06 ,02 ,08 ,04 ,12).
encomenda(31 ,08 ,03 ,09 ,75 ,50).
encomenda(32 ,09 ,01 ,11 ,25 ,30).
encomenda(33 ,10 ,04 ,08 ,16 ,15).
encomenda(34 ,13 ,01 ,04 ,12 ,07).
encomenda(35 ,14 ,02 ,07 ,83 ,04).
encomenda(36 ,15 ,01 ,08 ,04 ,20).
```

Entrega: Constituído por : *Id_entrega, Data, Id_encomenda, Classificação, Transporte, Preço, Tempo_Demorado.*

```
% entrega(id_entrega,Data,id_encomenda,classificacao,transporte,Preço,Tempo_demorado(horas))

entrega(01 ,date(2020,06,12) ,01 ,4.3 ,mota ,43 ,0.6).
entrega(02 ,date(2021,08,13) ,02 ,5.0 ,bicicleta ,23 ,2).
entrega(03 ,date(2021,01,23) ,03 ,4.6 ,carro ,52 ,0.5).
entrega(04 ,date(2020,06,12) ,04 ,5.0 ,bicicleta ,54 ,3).
entrega(05 ,date(2021,01,23) ,05 ,4.8 ,mota ,6 ,1).
entrega(06 ,date(2021,04,25) ,06 ,1.5 ,bicicleta , 24 ,8).
entrega(07 ,date(2020,04,12) ,07 ,3.2 ,bicicleta ,43 ,5.8).
entrega(08 ,date(2021,11,13) ,08 ,3.9 ,carro ,23 ,0.25).
entrega(09 ,date(2021,11,23) ,09 ,4.6 ,bicicleta ,52 ,3).
entrega(10 ,date(2020,12,12) ,10 ,4.5 ,carro ,54 ,0.32).
entrega(11 ,date(2021,01,23) ,11 ,4.0 ,carro ,6 ,0.25).
entrega(12 ,date(2021,02,25) ,12 ,4.5 ,mota , 24 ,2.25).
entrega(13 ,date(2020,09,12) ,13 ,4.3 ,mota ,43 ,2.5).
entrega(14 ,date(2021,08,13) ,14 ,3.9 ,bicicleta ,23 ,3).
entrega(15 ,date(2021,04,23) ,15 ,2.7 ,bicicleta ,52 ,6.2).
entrega(16 ,date(2020,06,13) ,16 ,4.5 ,mota ,54 ,1.25).
entrega(17 ,date(2020,06,13) ,17 ,4.9 ,carro ,6 ,0.05).
entrega(18 ,date(2021,04,21) ,18 ,4.5 ,carro , 24 ,0.65).
entrega(19 ,date(2020,03,18) ,19 ,2.5 ,mota ,43 ,9.2).
entrega(20 ,date(2021,08,17) ,20 ,3.9 ,mota ,23 ,0.52).
entrega(21 ,date(2021,01,14) ,21 ,2.0 ,bicicleta ,52 ,14).
entrega(22 ,date(2020,04,15) ,22 ,4.5 ,carro ,54 ,5.31).
entrega(23 ,date(2021,09,19) ,23 ,4.9 ,mota ,6 ,0.2).
entrega(24 ,date(2021,04,29) ,24 ,4.5 ,carro , 24 ,0.25).
```

Rua: Constituído por : *Nome_Rua, Freguesia, Cidade, Distância.*

```
% rua(Nome,Freguesia,Cidade,Distancia)

rua(jose_maria_ottoni ,nogueiro,braga,20).
rua(avenida_5_de_outubro ,serafao,fafe,10).
rua(rua_central,viade_baixo ,montalegre,50).
rua(santa_luzia,barcelos ,barcelos,15).
rua(sao_jorge,barcelos ,barcelos,35).
rua(duque_de_caxias ,nogueiro,braga,40).
rua(projetada ,fervidelas,montalegre,65).
rua(rui_barbosa,braga ,braga,25).
rua(santa_catarina,barcelos ,barcelos,15).
rua(boa_vista,santo_tirso ,porto,32).
rua(sao_luis ,nogueiro,braga,25).
rua(vinte_e_quatro ,viade_cima,ruivaes,45).
rua(sete_ceus,viade_baixo ,montalegre,50).
rua(beco_belo,barcelos ,barcelos,23).
rua(amorosa,barcelos ,barcelos,31).
rua(travessa_do_rio ,nogueiro,braga,25).
rua(santa_maria ,serafao,fafe,5).
rua(lamelas,fujacal ,braga,29).
rua(bairro_feliz,praca ,lisboa,320).
rua(augusta,padeirinhos,braga ,18).
rua(calcada_santana,vila_cova,fafe,70).
rua(avenida_paz,vila_moura,algarve,531).
rua(rua_barros, gualtar,braga,6).
rua(caranda,fujacal,braga,18).
```

Transporte: Constituído por : *Nome, Velocidade, Peso_Suportado.*

```
% transporte(nome,velocidade(km/h),peso(kg));

transporte(bicicleta ,10 ,5).
transporte(mota ,35 ,20).
transporte(carro ,25 ,100).
```

Observações

Na elaboração dos predicados, optou-se por associar ao predicado ‘rua’ a distância, ao invés desta ser colocada na entrega. Esta decisão foi tomada uma vez que, deste modo, tem-se o conhecimento prévio de qual será a distância, o que facilita bastante o trabalho. Assim, sabe-se não só qual a distância sabendo o cliente, como também há mais facilidade em perceber que veículo utilizar.

Para além disso, decidiu-se colocar o ID do estafeta na entrega, para que se consiga diferenciar mais facilmente as encomendas entregues e não entregues por estafeta.

Predicados Auxiliares

Predicado Evolução

O Predicado Evolução adiciona qualquer elemento à base de dados. A sua adição pode ser 'aceite' e o elemento é adicionado através do *assert*, caso contrário, o elemento é removido através do *retract.* Para verificar se o elemento em questão é ou não aceite, foi utilizado o *findall* que lista os invariantes relacionados com o elemento.

```
%Extensao do predicado evolucao:

evolucao(Termo):-
                findall(Invariante,+Termo::Invariante,Lista),
                insercao(Termo),
                teste(Lista).

insercao(Termo):-
                assert(Termo).
insercao(Termo):-
                retract(Termo),!,fail.
```

Predicado Involução

O Predicado Involução é precisamente o contrário do Predicado Evolução. Assim, para diferenciar os invariantes deste predicado e do anterior, utiliza-se o sinal '-'.

```
%Extensao do predicado involucao:

involucao(Termo):-
        findall(Invariante,-Termo::Invariante,Lista),
        remocao(Termo),
        teste(Lista).

remocao(Termo):-
    retract(Termo).

remocao(Termo):-
    assert(Termo),!,fail.
```

Invariantes

Os Invariantes são restrições desenvolvidas quando é necessário adicionar ou retirar algo da base de conhecimento.

Assim, foram criados Invariantes de acordo com restrições específicas:

- Não permite criar um estafeta com um ID já existente:

```
+estafeta(Id,Nome,Idade) :: (findall((Id),(estafeta(Id,_,_)),S),
    comprimento(S,N), N == 1).
```

- Não permite criar um estafeta com um nome e idade igual a outro:

```
+estafeta(Id,Nome,Idade) :: (findall((Nome,Idade),(estafeta(_,Nome,Idade)),S),
    comprimento(S,N), N == 1).
```

- Não permite criar um cliente com um ID já existente:

```
+cliente(Id,Nome,Idade,Rua) :: (findall((Id),(cliente(Id,_,_,_)),S),
    comprimento(S,N), N == 1).
```

- Não permite criar um cliente com um nome, idade e rua igual ao de outro:

```
+cliente(Id,Nome,Idade,Rua) :: (findall((Nome,Idade,Rua),(cliente(_,Nome,Idade,Rua)),S),
    comprimento(S,N), N == 1).
```

- Não permite criar uma encomenda com um ID já existente:

```
+encomenda(Id,Nome,Idade,Rua) :: (findall((Id),(encomenda(Id,_,_,_)),S),
    comprimento(S,N), N == 1).
```

- Não permite criar uma entrega com um ID já existente:

```
+entrega(Id,Data,Id_encomenda,Classificacao,Transporte,Preço,Tempo_demorado) ::
(findall((Id),(entrega(Id,_,_,_,_,_,_)),S),comprimento(S,N), N == 1).
```

- Não permite criar uma entrega com o ID da encomenda igual a outra entrega:

```
+entrega(Id,Data,Id_encomenda,Classificacao,Transporte,Preço,Tempo_demorado) ::
(findall((Id_encomenda),(entrega(_,Id_encomenda,_,_,_,_,_)),S),comprimento(S,N), N == 1).
```

- Não permite remover um estafeta com encomendas referentes a ele:

```
-estafeta(Id,Nome,Idade) :: (findall(Id,encomenda(_,_,Id,_,_,_),S),comprimento(S,N),N == 0).
```

- Não permite remover um cliente com encomendas referentes a ele:

```
-cliente(Id,Nome,Idade,Rua) :: (findall(Id,encomenda(_,Id,_,_,_,_),S),comprimento(S,N),N == 0).
```

Funcionalidades

Para todas as Funcionalidades foram utilizados vários predicados auxiliares, dos quais se destaca o predicado *findall*, cuja função é percorrer a base de conhecimento, retornando qualquer elemento(s) ou até mesmo variável(is) que lá se encontrem.

- **Registar/Remover estafetas**

```
%Extensao do predicado registar_estafeta:

registar_estafeta(Id,Nome,Idade):-evolucao(estafeta(Id,Nome,Idade)).
remover_estafeta(Id,Nome,Idade):-involucao(estafeta(Id,Nom,Idade)).
```

- **Registar/Remover clientes**

```
%Extensao do predicado registar_cliente:

registar_cliente(Id,Nome,Idade,Rua):-evolucao(cliente(Id,Nome,Idade,Rua)).
remover_cliente(Id,Nome,Idade,Rua):-involucao(cliente(Id,Nom,Idade,Rua)).
```

- **Registar/Remover encomendas**

```
%Extensao do predicado registar_encomenda:

registar_encomenda(Id,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume):-evolucao(encomenda(Id,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume)).
remover_encomenda(Id,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume):-involucao(encomenda(Id,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume)).
```

- **Registar/Remover entregas**

```
%Extensao do predicado registar_entrega:

registar_entrega(Id_entrega,Data,Id_encomenda,Classificacao,Id_transporte,Preco,Tempo_demorado):-
  evolucao(entrega(Id_entrega,Data,Id_encomenda,Classificacao,Id_transporte,Preco,Tempo_demorado)).
remover_entrega(Id_entrega,Data,Id_encomenda,Classificacao,Id_transporte,Preco,Tempo_demorado):-
  involucao(entrega(Id_entrega,Data,Id_encomenda,Classificacao,Id_transporte,Preco,Tempo_demorado)).
```

- **Identificar qual o estafeta mais ecológico**

  Foram percorridos todos os estafetas, comparando qual deles utilizou mais vezes o veículo mais ecológico. Para este predicado, foram utilizadas os predicados auxiliares:

  - *maisfrequente*: Dada uma lista, retorna o(s) elemento(s) mais frequente(s);
  - *comprimento*: Dada uma lista, retorna o comprimento da mesma.

```
%Extensao do predicado estafetaecologico:

estafetaecologico(X):-findall((Id_estafeta,Nome,Idade),(entrega(_,_,Id_encomenda,_,bicicleta,_,_),
  encomenda(Id_encomenda,_,Id_estafeta,_,_,_),estafeta(Id_estafeta,Nome,Idade)),Z),
comprimento(Z,A),A=\=0,!,maisfrequente(Z,X),write('Utilizou/Utilizaram mais vezes bicicleta').
estafetaecologico(X):-findall((Id_estafeta,Nome,Idade),(entrega(_,_,Id_encomenda,_,mota,_,_),
  encomenda(Id_encomenda,_,Id_estafeta,_,_,_),estafeta(Id_estafeta,Nome,Idade)),Z),
comprimento(Z,A),A=\=0,!,maisfrequente(Z,X),write('Utilizou/Utilizaram mais vezes mota').
estafetaecologico(X):-findall((Id_estafeta,Nome,Idade),(entrega(_,_,Id_encomenda,_,carro,_,_),
  encomenda(Id_encomenda,_,Id_estafeta,_,_,_),estafeta(Id_estafeta,Nome,Idade)),Z),
comprimento(Z,A),A=\=0,!,maisfrequente(Z,X),write('Utilizou/Utilizaram mais vezes carro').
```

- **Identificar qual foi o estafeta responsável por uma entrega**

  Dado o ID de uma entrega, é retornado o estafeta que efetuou a sua entrega.

```
%Extensao do predicado estafetaresponsavel:

estafetaresponsavel(Id_entrega,S):-findall((Id_estafeta,Nome,Idade),
  (entrega(Id_entrega,_,Id_encomenda,_,_,_,_),encomenda(Id_encomenda,_,Id_estafeta,_,_,_),
    estafeta(Id_estafeta,Nome,Idade)),S),
  write('Ordem do predicado estafeta: Id_estafeta,Nome,Idade').
```

- **Identificar quais foram os clientes servidos por um estafeta**

  Dado o ID de um estafeta, é retornada a lista de clientes servidos pelo mesmo.

```
%Extensao do predicado clientesservidos:

clientesservidos(Id_estafeta,X):-findall((Id_cliente,Nome,Idade,Rua),(entrega(_,_,Id_encomenda,_,_,_,_),
  encomenda(Id_encomenda,Id_cliente,Id_estafeta,_,_,_),cliente(Id_cliente,Nome,Idade,Rua)),X),
write('Ordem do predicado cliente: Id_cliente,Nome,Idade,Rua').
```

- **Calcular o dinheiro ganho pela Green Distribution num determinado dia:**

Dado um determinado dia, é calculada a soma de todas as entregas feitas. Para este predicado, foi utilizado o seguinte predicado auxiliar:

- *soma:* Dada uma lista, retorna a soma de todos os seus elementos.

```
%Extensao do predicado dinheironodia:

dinheironodia(Data,S):-
findall(Preço,entrega(_,Data,_,_,_,Preço,_),R),soma(R,S).
```

- **Identificar qual é a rua mais movimentada pela Green Distribution:**

Verificando todas as entregas, vemos qual do(s) cliente(s) receberam mais encomendas, retornando assim a rua do(s) mesmo(s). Para este predicado foi utilizado o seguinte predicado auxiliar:

- *maisfrequente*: Dada uma lista, retorna o(s) elemento(s) mais frequente(s).

```
%Extensao do predicado ruamaismovimentada:

ruamaismovimentada(X):-findall((Nome_Rua,Freguesia,Cidade,Distancia),(entrega(_,_,Id_encomenda,_,_,_,_),
  encomenda(Id_encomenda,Id_cliente,_,_,_,_),cliente(Id_cliente,_,_,Nome_Rua),
  rua(Nome_Rua,Freguesia,Cidade,Distancia)),Z),maisfrequente(Z,X).
```

- **Calcular a classificação média de um estafeta:**

Dado o ID de um estafeta, foram percorridas todas as suas entregas e calculou-se a média das classificações. Para este predicado, foi utilizado o seguinte predicado auxiliar:

- *media*: Dada uma lista, retorna a sua média.

```
%Extensao do predicado classestafeta:

classestafeta(Id_estafeta,X):-
findall((Classificacao),(encomenda(Id_encomenda,_,Id_estafeta,_,_,_),
  entrega(_,_,Id_encomenda,Classificacao,_,_,_)),Z),media(Z,X).
```

- **Identificar o número total de entregas, pelos diferentes meios de transporte, num determinado intervalo de tempo:**

Verifica todas as entregas, restringe o intervalo de tempo e separa-as por meios de transporte, retornando uma lista com 3 elementos para cada meio. Para este predicado, foram utilizadas os predicados auxiliares:

- *auxiliar*: Dado um intervalo de tempo e um meio de transporte, retorna quantas vezes o mesmo foi utilizado durante o intervalo de tempo.
- *menores*: Dada uma lista de datas e uma data, remove da lista todas as datas maiores.
- *maiores*: Dada uma lista de datas e uma data, remove da lista todas as datas menores.
- *comprimento*: Dada uma lista, retorna o comprimento da mesma.
- *add_tail:* Dada uma lista e um elemento, adiciona o elemento à cauda da lista.

```
%Extensao do predicado totalentregasportransporte:

totalentregasportransporte(date(Y,M,D),date(Y1,M1,D1),K):- auxiliar(date(Y,M,D),date(Y1,M1,D1),bicicleta,Z)
,add_tail([],Z,W),auxiliar(date(Y,M,D),date(Y1,M1,D1),mota,A),add_tail(W,A,V),
auxiliar(date(Y,M,D),date(Y1,M1,D1),carro,B),add_tail(V,B,K),
write('Numero de entregas de bicicleta, mota e carro respetivamente').

auxiliar(date(Y,M,D),date(Y1,M1,D1),X,Z):-
findall(Data,entrega(_,Data,_,_,X,_,_),V),maiores(V,date(Y,M,D),H),
menores(H,date(Y1,M1,D1),A),comprimento(A,Z).
```

- **Identificar o número total de entregas pelos estafetas, num determinado intervalo de tempo:**

Verifica todas as entregas, restringe o intervalo de tempo e separa-as por todos os estafetas, retornando uma lista cujo tamanho é igual ao número de estafetas. Para este predicado, foram utilizados os predicados auxiliares:

- *auxiliar2:* Dado um intervalo de tempo, um acumulador e a quantidade de estafetas existentes, percorre todos os estafetas e retorna uma lista em que cada elemento é o número de entregas por cada estafeta, ordenadamente.
- *menores*: Dada uma lista de datas e uma data, remove da lista todas as datas maiores.
- *maiores*: Dada uma lista de datas e uma data, remove da lista todas as datas menores.
- *comprimento*: Dada uma lista, retorna o comprimento da mesma.
- *add_tail*: Dada uma lista e um elemento, adiciona o elemento à cauda da lista.

```
%Extensao do predicado totalentregasporestafeta:

totalentregasporestafeta(X,Y,Z):- findall(Id_estafeta,estafeta(Id_estafeta,_,_),A),
comprimento(A,H),auxiliar2(X,Y,1,H,[],Z),write('Cada elemento é o numero de entregas por estafeta, começando pelo primeiro ordenadamente').

auxiliar2(J,L,H,X,Z,Z):-H>X.
auxiliar2(J,L,H,X,A,Z):-
findall(Data,(entrega(_,Data,Id_encomenda,_,_,_,_),encomenda(Id_encomenda,_,H,_,_,_)),C),
maiores(C,J,K),menores(K,L,B),comprimento(B,S),add_tail(A,S,N),M is H+1,auxiliar2(J,L,M,X,N,Z).
```

- **Calcular o número de encomendas entregues e não entregues pela Green Distribution, num determinado intervalo de tempo:**

Calcula as entregas realizadas num intervalo de tempo, subtrai com o número de encomendas total, e retorna uma lista com dois elementos, o número de encomendas entregues e não entregues, respetivamente. Para este predicado, foram utilizados os predicados auxiliares:

- *miniaux:* Realiza o trabalho do predicado original, mas adiciona uma variável, que é uma lista, para facilitar a resolução.
- *comprimento*: Dada uma lista, retorna o comprimento da mesma.
- *add_tail:* Dada uma lista e um elemento, adiciona o elemento à cauda da lista.

```
%Extensao do predicado entreguesenaoentregues:

entreguesenaoentregues(X,Y,Z):-miniaux(X,Y,[],Z),write('Encomendas entregues e não entregues, respetivamente').

miniaux(X,Y,W,Z):- findall(Data,entrega(_,Data,Id_encomenda,_,_,_,_),S),comprimento(S,N),
  findall(Id_encomenda,encomenda(Id_encomenda,_,_,_,_,_),M),comprimento(M,G),
  V is G-N,add_tail(W,N,H),add_tail(H,V,Z).
```

- **Calcula o peso total transportado por estafeta, num determinado dia:**

Restringe as entregas realizadas apenas para um intervalo de tempo específico, e percorre-as, criando uma lista com tamanho igual ao número de estafetas, preenchendo-a com o peso total transportado por cada um. Para este predicado foram utilizados os predicados auxiliares:

- *funcao & funcao2:* Predicados auxiliares que são apenas utilizados para facilitar o trabalho deste predicado em específico.
- *add_tail:* Dada uma lista e um elemento, adiciona o elemento à cauda da lista.
- *soma:* Dada uma lista, retorna a soma de todos os seus elementos.

```
%Extensao do predicado pesototal:

pesototal(Data,X):- findall(Id_estafeta,estafeta(Id_estafeta,_,_),Z),
comprimento(Z,H),funcao(Data,1,H,[],X).


funcao(Data,X,Q,Z,Z):- Q < X.
funcao(Data,X,Y,Q,Z):- funcao2(Data,X,A),add_tail(Q,A,G),V is X+1,funcao(Data,V,Y,G,Z).

funcao2(Data,X,Z):- findall(Peso,(entrega(_,Data,Id_encomenda,_,_,_,_),
  encomenda(Id_encomenda,_,X,_,Peso,_)),V),soma(V,Z).
```

- **Todos estafetas/clientes/encomendas/entregas/ruas:**

Predicados que mostram todos os elementos existentes na base de conhecimento relativamente a um predicado.

1. **Todos estafetas:**

```
%Extensao do predicado todosestafetas:

todosestafetas(S):-
findall((Id,Nome,Idade),estafeta(Id,Nome,Idade),S),
write('Ordem do predicado estafeta: Id_estafeta,Nome,Idade').
```

2. **Todos clientes:**

```
%Extensao do predicado todosclientes:

todosclientes(S):-
findall((Id_cliente,Nome,Idade,Rua),cliente(Id_cliente,Nome,Idade,Rua),S),
write('Ordem do predicado cliente: Id_cliente,Nome,Idade,Nome_Rua').
```

3. **Todas encomendas:**

```
%Extensao do predicado todasencomendas:

todasencomendas(S):-
findall((Id_encomenda,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume),
  encomenda(Id_encomenda,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume),S),
write('Ordem do predicado encomenda: Id_encomenda,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume').
```

4. **Todas entregas:**

```
%Extensao do predicado todasentregas:

todasentregas(S):-
findall((Id_entrega,Data,Id_encomenda,Classificacao,Transporte,Preço,Tempo_demorado),
entrega(Id_entrega,Data,Id_encomenda,Classificacao,Transporte,Preço,Tempo_demorado),S),
write('Ordem do predicado entrega:Id_entrega,Data,Id_encomenda,Classificacao,Transporte,Preço,Tempo_demorado').
```

5. **Todas ruas:**

```
%Extensao do predicado todasruas:

todasruas(S):-
findall((Nome_Rua,Freguesia,Cidade,Distancia),rua(Nome_Rua,Freguesia,Cidade,Distancia),S),
write('Ordem do predicado rua: Nome_Rua,Freguesia,Cidade,Distancia').
```

- **Identifica todas as entregas realizadas por um estafeta:**

Verifica todas as entregas com o ID do estafeta dado, retornando assim uma lista com todas as que foram entregues pelo mesmo.

```
%Extensao do predicado estafetadeliver:

estafetadeliver(Id_estafeta,S):-
findall((Id_entrega,Data,Id_encomenda,Classificacao,Transporte,Preço,Tempo_demorado),
(encomenda(Id_encomenda,_,Id_estafeta,_,_,_),
entrega(Id_entrega,Data,Id_encomenda,Classificacao,Transporte,Preço,Tempo_demorado)),S),
write('Ordem do predicado entrega: Id_entrega,Data,Id_encomenda,Classificacao,Transporte,Preço,Tempo_demorado').
```

- **Identifica a encomenda dado ID da entrega:**

```
%Extensao do predicado entregainfoencomenda:

entregainfoencomenda(Id_entrega,S):-
findall((Id_encomenda,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume),
  (entrega(Id_entrega,_,Id_encomenda,_,_,_),encomenda(Id_encomenda,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume)),S),
write('Ordem do predicado encomenda: Id_encomenda,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume').
```

- **Fornece informação de qualquer predicado dado o seu ID:**

1. **Cliente Info:**

```
%Extensao do predicado clienteinfo:

clienteinfo(Id_cliente,S):-
findall((Id_cliente,Nome,Idade,Rua),cliente(Id_cliente,Nome,Idade,Rua),S),
write('Ordem do predicado cliente: Id_cliente,Nome,Idade,Rua').
```

2. **Estafeta Info:**

```
%Extensao do predicado estafetainfo:

estafetainfo(Id_estafeta,S):-
findall((Id_estafeta,Nome,Idade),estafeta(Id_estafeta,Nome,Idade),S),
write('Ordem do predicado estafeta: Id_estafeta,Nome,Idade').
```

3. **Encomenda Info:**

```
%Extensao do predicado encomendainfo:

encomendainfo(Id_encomenda,S):-
findall((Id_encomenda,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume),
  encomenda(Id_encomenda,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume),S),
write('Ordem do predicado encomenda: Id_encomenda,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume').
```

4. **Entrega info:**

```
%Extensao do predicado entregainfo:

entregainfo(Id_entrega,S):-
findall((Id_entrega,Data,Id_encomenda,Classificacao,Transporte,Preço,Tempo_demorado),
  entrega(Id_entrega,Data,Id_encomenda,Classificacao,Transporte,Preço,Tempo_demorado),S),
write('Ordem do predicado entrega: Id_entrega,Data,Id_encomenda,Classificacao,Transporte,Preço,Tempo_demorado').
```

- **Verifica todas as encomendas associadas a um cliente:**

```
%Extensao do predicado encomendascliente:

encomendascliente(Id_cliente,S):-
findall((Id_encomenda,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume),
  encomenda(Id_encomenda,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume),S),
write('Ordem do predicado encomenda: Id_encomenda,Id_cliente,Id_estafeta,Prazo_pedido,Peso,Volume').
```

- **Identifica o estafeta que realizou uma entrega:**

```
%Extensao do predicado estafetadaentrega:

estafetadaentrega(Id_entrega,X):-
findall((Id_estafeta,Nome,Idade),(entrega(Id_entrega,_,Id_encomenda,_,_,_,_),
  encomenda(Id_encomenda,_,Id_estafeta,_,_,_),estafeta(Id_estafeta,Nome,Idade)),X).
```

- **Dada uma encomenda, verifica qual o melhor veículo a ser utilizado:**

Compara os pesos suportados por todos os veículos e a velocidade, consoante o tempo pedido pelo cliente, priorizando sempre o veículo mais ecológico.

```
%Extensao do predicado melhorveiculo:

melhorveiculo(Id_encomenda,bicicleta):-findall(Peso,encomenda(Id_encomenda,_,_,_,Peso,_),X),
cabeca(X,Y),5>=Y,findall(Prazo_pedido,encomenda(Id_encomenda,_,_,Prazo_pedido,_,_),Z),cabeca(Z,W),
findall(Distancia,(encomenda(Id_encomenda,Id_cliente,_,_,_,_),cliente(Id_cliente,_,_,Nome_Rua),
rua(Nome_Rua,Freguesia,Cidade,Distancia)),A),cabeca(A,B),Z>=B/10,write('O melhor veiculo seria bicicleta').

melhorveiculo(Id_encomenda,mota):-findall(Peso,encomenda(Id_encomenda,_,_,_,Peso,_),X),
cabeca(X,Y),20>=Y,findall(Prazo_pedido,encomenda(Id_encomenda,_,_,Prazo_pedido,_,_),Z),cabeca(Z,W),
findall(Distancia,(encomenda(Id_encomenda,Id_cliente,_,_,_,_),cliente(Id_cliente,_,_,Nome_Rua),
rua(Nome_Rua,Freguesia,Cidade,Distancia)),A),cabeca(A,B),Z>=B/35,write('O melhor veiculo seria mota').

melhorveiculo(Id_encomenda,carro):-findall(Peso,encomenda(Id_encomenda,_,_,_,Peso,_),X),
cabeca(X,Y),100>=Y,findall(Prazo_pedido,encomenda(Id_encomenda,_,_,Prazo_pedido,_,_),Z),cabeca(Z,W),
findall(Distancia,(encomenda(Id_encomenda,Id_cliente,_,_,_,_),cliente(Id_cliente,_,_,Nome_Rua),
rua(Nome_Rua,Freguesia,Cidade,Distancia)),A),cabeca(A,B),Z>=B/25,write('O melhor veiculo seria carro').


cabeca([H],H).
```

- **Calcula qual é o melhor preço para uma encomenda:**

Através de valores estipulados pelo grupo, calcula o preço de uma encomenda, consoante o veículo utilizado e a distância à rua do cliente.

```
%Extensão do predicado melhorpreco:

melhorpreco(Id_encomenda,X):-findall(Distancia,(encomenda(Id_encomenda,Id_cliente,_,_,_,_),
  cliente(Id_cliente,_,_,Nome_Rua),rua(Nome_Rua,Freguesia,Cidade,Distancia)),Z),cabeca(Z,Y),
melhorveiculo(Id_encomenda,bicicleta),X is Y*0.3.
melhorpreco(Id_encomenda,X):-findall(Distancia,(encomenda(Id_encomenda,Id_cliente,_,_,_,_),
  cliente(Id_cliente,_,_,Nome_Rua),rua(Nome_Rua,Freguesia,Cidade,Distancia)),Z),cabeca(Z,Y),
melhorveiculo(Id_encomenda,mota),X is Y*0.6.
melhorpreco(Id_encomenda,X):-findall(Distancia,(encomenda(Id_encomenda,Id_cliente,_,_,_,_),
  cliente(Id_cliente,_,_,Nome_Rua),rua(Nome_Rua,Freguesia,Cidade,Distancia)),Z),cabeca(Z,Y),
melhorveiculo(Id_encomenda,carro),X is Y*1.2.
```

Conclusão

O trabalho proporcionou uma melhoria bastante significativa em relação aos sistemas de representação de conhecimento e raciocínio.

Pode-se observar que, não só todos os requisitos propostos foram cumpridos, como também foram adicionadas outras funcionalidades bastante úteis, uma vez que promovem uma maior versatilidade do projeto.

A realização do trabalho foi bastante enriquecedora, pois permitiu aprofundar e consolidar conceitos aprendidos em contexto sala de aula e consolidar e a adquirir competências que serão uma mais valia para o futuro.

Por fim, importa salientar que existem outras funcionalidades que seriam interessantes desenvolver no sentido de complementar e otimizar o sistema em questão, nomeadamente a implementação de uma funcionalidade que, colocando um determinado valor, indicasse que tipo de transporte estaria disponível e que distância seria possível percorrer, mediante esse preço.