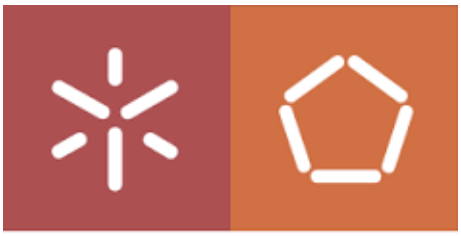
Projeto-Relatório

2 de dezembro de 2021



Universidade do Minho
Escola de Engenharia

grupo 16:
Cláudia Silva, a93177
João Mendes, a93256
André Vaz, a93221
Tomás Ogando, a93174

Conteúdo

1 Introdução

Na primeira fase deste trabalho pretendeu-se desenvolver um sistema de representação de conhecimento e raciocínio com capacidade para caracterizar um universo de discurso na área da logística de distribuição de encomendas. Para tal tivemos de desenvolver um sistema de encomendas para uma empresa denominada por *Green Distribution* que tem como objetivo privilegiar a ecologia acima de tudo. Cada estafeta desta empresa foi associado a diversas encomendas podendo utilizar três possíveis transportes. Tendo isto em consideração, elaboramos diversas funções em *Prolog* que, utilizando a base de dados criada por nós, dão-nos acesso a várias informações.

Com isto, decidimos optar para uma melhor pratica dividir o projeto criando dois ficheiros *Prolog*: *basedados.pl*, onde estão presentes os predicados e conhecimento que sustentam o trabalho e *queries.pl* onde estão implementadas todas as funcionalidades propriamente ditas.

2 Base de Conhecimento

Nesta primeira fase começamos por definir a base de dados. Consideramos 6 estafetas (Cátia, Carlos, Sheila, Ilídio, Filipe e Roger) e 6 clientes (Ogando, Paulo, Marlene, Susana, Jéssica e Sandro) cada um tendo uma rua associada e, consequentemente, cada uma das ruas uma freguesia associada. Relacionamos também um índice de poluição a cada veículo, desta forma pudemos mais facilmente descobrir quem utiliza mais vezes o meio de transporte mais ecológico, sendo util para *query* 1. De facto, consideramos que o nível de poluição da bicicleta era 0 (uma vez que ela não causa poluição), o nível de poluição do carro era 2 e da mota 1. Desta maneira, assumimos que uma viagem carro polui tanto quanto duas de mota.

Em relação ao conteúdo das encomendas, decidimos adotar 4 tipos diferentes. Cada um dos tipo tem um peso e um volume associados. Há a possibilidade de transportar livros, roupa, um computador e uma bicicleta. Isto permitiu-nos realizar com mais facilidade a última *query* proposta em que tinhamos que calcular o peso total transportado num dia.

Além disso, criamos o caminho, que nos indica várias informações. Na verdade, através do facto *caminho* sabemos quem foi o cliente, o estafeta que vai realizar a entrega, o veículo que vai ser utilizado, a data e hora que foi encomendado, o que leva a encomenda, o preço de transporte, o espaço de tempo em que o cliente a quer receber e o número de série da mesma. Dessa forma, ao associar o número de série a cada encomenda garantimos que não há uma repetição da mesma entrega (assim se um cliente realizar duas encomendas seguidas garantimos que são distintas). Foi também criada a confirmação dos factos para conseguirmos comparar as encomendas ainda por completar. Com estas confirmações temos acesso aos dados de cada encomenda, nomeadamente, a data de entrega da mesma, o *ranking* e o tempo que demorou a ser entregue. Posteriormente, ajudam-nos a confirmar resultados sobre que estafeta fez determinada entrega.

Para as *queries* 7, 8 e 9 em relação aos intervalos de tempo decidimos considerar dias para obter o resultado, comparando a data de saída com a data de chegada.

Fomos encorajados a incluir novas funcionalidades neste projeto e decidimos aceitar o desafio desenvolvendo duas novas funcionalidades. A primeira adição ao projeto trata-se de uma forma de descobrir que entregas foram feitas numa determinada hora. Utilizando funções e métodos de resolução aplicados previamente, conseguimos implementar esta query. Além disso, também criamos uma forma de descobrir quais as encomendas entregues a horas e das encomendas em falta.

3 Queries

3.1 Query 1

Na *query* 1 tinhamos como objetivo obter o estafeta que utilizou mais vezes um veiculo mais ecologico, sendo este a bicicleta.

Deste jeito, começamos por definir o predicado *query1_aux* que recebe um estafeta e devolve o número de vezes que o mesmo utilizou como meio de transporte uma bicicleta. Assim, tendo a lista de todos os estafetas podemos calcular o número de vezes que cada um deles utilizou a bicicleta, sendo isto feito através da *query1_tuple*. Tendo os valores para cada estafeta falta apenas orderar a lista de forma a que o estafeta com maior valor fique em primeiro, sendo isto realizado na função *compare_by_second*, podendo então concluir que o estafeta mais ecológico é o que estiver em primeiro na lista.

```
?- query1(X).
X = carlos .
```

3.2 Query 2

Para identificar que estafetas entregaram determinada(s) encomenda(s) a um determinado cliente optamos por criar duas funções.

Primeiramente, o predicado *query2_aux* que dado um cliente e uma encomenda vai descobrir o estafeta que a realizou e verificar o número de série. De seguida, confirma se essa encomenda foi entregue .Posteriormente, averigua se se trata de um estafeta. Na função *query2*, com o auxilio da função *findall* conseguimos aplicar a *query2_aux* a todos os caminhos.

```
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- query2(marle,encomenda(livros,1,5),Z).
Z = [carlos].

?- query2(paulo,encomenda(livros,1,5),Z).
Z = [].

?- query2(p,encomenda(livros,1,5),Z).
Z = [].

?- query2(paulo,encomenda(bicicleta,12,2800),Z).
Z = [ilidio].
```

3.3 Query 3

Para esta query *query* foi necessário identificar os clientes que foram servidos por um determinado estafeta dado pelo utilizador. Inicialmente, verifica se existe alguma entrega feita pelo estafeta a um determinado cliente através da *query3_aux*, que nos vai permitir depois na *query3* desenvolver uma lista com os clientes.

```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.2.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- query3(carlos,X).
X = [susana, marlene, marle].

?- query3(catia,X).
X = [jessica, paulo].
```

3.4 Query 4

A estratégia usada para saber o valor faturado pela *Green Distribution* num determinado dia consiste no desenvolvimento de uma auxiliar, *soma*, que tem como objetivo final somar todos os elementos da lista. Assumimos que o cliente paga mal faça a encomenda, assim sendo trabalhamos com o facto *caminho* para verificar se existe o dia e o preço inseridos pelo utilizador.

Por fim, desenvolvemos a *query4* com o objetivo de identificar o valor calculado, através da auxiliar soma e também pelo *findall* que nos permitiu juntar tudo numa lista para que a soma fosse possivel.

```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.0)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- query4(data(3,12,2021),X).
X = 0.

?- query4(data(24,11,2021),X).
X = 14.

?- query4(data(2,12,2021),X).
X = 96.
```

3.5 Query 5

Para a resolução desta *query* decidimos desenvolver várias auxiliares que ajudassem no processo para identificar as zonas com maior volume de entregas por parte da empresa *Green Distribution*. Assim sendo, numa primeira fase, desenvolvemos *query5_aux2* que agrupa numa lista todas as ruas existentes, através do *findall*. De seguida desenvolvemos uma auxiliar que tem o intuito de procurar uma dada rua numa lista com as ruas todas e agrupar o respetivo nome da rua consoante as vezes que aparece na lista, *procura*. A partir desta, conseguimos desenvolver a *query_comp* que organiza as ruas pela cabeça da lista. Através do desenvolvimento da *compare_by_second2* que compara de forma descrescente e da *query_comp*, como já foi referida, elaboramos a *query5_aux1* que organiza a lista de ruas de forma descrescente consoante o número de vezes que aparecem.

Por fim, a *query5* devolve a cabeça da lista que corresponde ao local com maior volume de entregas.

```
?- query5(Rua).
Rua = da_Torre .
```

3.6 Query 6

A resolução desta *query* consiste em calcular a classificação média de satisfação de clientes num determinado estafeta. Assim sendo, decidimos fazer uma lista com o número de viagens que determinado estafeta fez, por isso usamos o *length*. Para calcular a soma de todas as avaliações feita aos estafetas, agrupamos a avaliação numa lista e somamos. Por último, tendo o número de viagens realizadas por estafeta e a soma de todas as avaliações dessas viagens dadas pelos clientes, calculamos a média como é pedido.

```
?- query6_media(carlos,X).
X = 3.6666666666666665.

?- query6_media(sheila,X).
X = 2.

?- query6_media(catia,X).
X = 3.5.
```

3.7 Query 7

Para identificar o número total de entregas realizadas pelos diferentes meios de transporte, num determinado intervalo de tempo optamos por criar uma função auxiliar que converte uma data para dias, a função *convertDatatodays*. Posteriormente, é usada na função *comp* que compara uma data e verifica se essa se encontra no intervalo indicado. Posto isto, a função *aux* e *query* 7 percorrem o facto *confirmacao* "procurando"pelo veiculo indicado e verificando se a data se encontra entre o intervalo indicado, caso se encontre é acrescentada a uma lista que subconsequentemente é somado o tamanho. Para esta *query* é necessário que seja inserida primeiro a data mais antiga e só depois a data mais recente.

```
?- query7(data(5,10,2020),data(2,12,2021),bicicleta,C).
C = 3.

?- query7(data(5,10,2020),data(2,12,2021),carro,C).
C = 1.

?- query7(data(5,10,2020),data(2,12,2021),mota,C).
C = 2.
```

3.8 Query 8

Nesta *query*, percorrendo o facto *confirmacao* e comparando a data de cada um usando a função auxiliar *comp*, usada na *query* 7 conseguimos identificar o número total de entregas pelos estafetas, num determinado intervalo de tempo. Para esta *query* é necessário que seja inserida primeiro a data mais antiga e só depois a data mais recente.

```
For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- query8(data(5,10,2020),data(2,12,2021),C).
C = 6.

?- query8(data(5,10,2020),data(3,12,2021),C).
C = 11.

?- query8(data(24,11,2021),data(3,12,2021),C).
C = 7.
```

3.9 Query 9

A query 9 tem como objetivos calcular o número de encomendas entregues e não entregues pela *Green Distribution*, num determinado período de tempo. Nesta *query*, utilizamos a *query* 8 como auxiliar uma vez que a *query* 8 já identifica o número total de entregas pelos estafetas num determinado intervalo de tempo. Posteriormente, apenas contamos quantos factos *caminho* existem e subtraimos as que foram entregues naquele intervalo de tempo ao total.

```
?- query9(data(2,12,2021),data(2,12,2021),X).
X =  (2, 10).

?- query9(data(3,12,2021),data(3,12,2021),X).
X =  (5, 7).

?-
```

3.10 Query 10

Para o calculo do peso total transportado por um determinado estafeta num certo dia decidimos desenvolver duas funções.

Primeiramente, criamos uma função auxiliar, *query_10* que tem como objetivo verificar se existe alguma encomenda que foi realizada consoante o estafeta ,dia e peso, através do facto *caminho* e do facto *confirmacao*.

De seguida, baseamo-nos no mesmo processo de desenvolvimento feito para a *query4*, que consiste em armazenar o peso numa lista através do *findall* e, consequentemente, somar todos os elementos da lista para devolver o peso total num determinado dia.

```
?- query10(catia,data(2,12,2021),X).
X = 18.

?- query10(carlos,data(4,10,2020),X).
X = 0.

?- query10(sheila,data(2,12,2021),X).
X = 0.

?- query10(sheila,data(3,12,2021),X).
X = 6.

?-
```

3.11 Query Adicionais

Nesta secção iremos abordar algumas *queries* que decidimos elaborar.

3.11.1 Query Teste

A query teste verifica quantas encomendas foram, de facto, entregues.Ou seja,indica quantos caminhos tem a confirmação (usando o número de série).

```
?- queryteste(X).
X = 11.
```

3.11.2 Query 11

Elaboramos um facto que determinasse o número de entregas que ocorreram numa dada hora escolhida pelo utilizador. Para isso , através do *findall* desenvolvemos uma lista com todas as encomendas que ocorreram numa dada hora e por intermédio do *length* determinamos o tamanho da lista que corresponde ao número de entregas realizadas.

```
?- query11(hora(19,00),X).
X = 2.
```

3.11.3 Query 12

Com o objetivo de obter uma lista com as diferentes entregas realizadas atempadamente, decidimos converter o tempo do facto *caminho* e o tempo do facto *confirmação* para horas, da seguinte maneira:

```
converteTempoHoras(tempo(M,D,H),X):-
    X is H+(D*24)+(M*30*24).
```

Com os tempos convertidos, podemos determinar se o tempo de confirmação é menor ou igual ao tempo do caminho, se tal o for então a entrega foi realizada atempadamente.

Desta forma conseguimos concluir que as entregas que cumprem tal requisito são as seguintes:

```
?- query12(X).
X = [2001, 2003, 2006, 2007, 2008, 2010, 2011, 2012].
```

3.11.4 Query 13

O invariante criado apenas deixa inserir um caminho se o número de série ainda não existir.

3.11.5 Query 14

Esta *query* teve origem a partir da *query1*, onde tem como resultado o estafeta que foi mais ecológico ao fim das entregas todas.

```
?- query14(X).
X = carlos .
```

4 Conclusão

Em arremate, este projeto permitiu-nos adquirir competências fulcrais no desenvolvimento de sistemas com capacidades de caracterizar um universo na área da logística de distribuição de encomendas. A utilização de uma linguagem como o *Prolog* trouxe enúmeras vantagens na realização do projeto pois trata se de uma linguagem orientada à funcionalidade puramente lógica. Ademais, foi imprescindível o uso da nossa própria base de dados pois, permitiu uma maior liberdade de como abordar os problemas propostos. É opinião do grupo que esta fase foi concluída com sucesso.