

Relatório trabalho prático TIA

-Técnicas de Inteligência artificial

Equipa:

a72349 - Ricardo Oliveira

a92874 - Francisca Mieiro

a92887 - Joaquim Castro

Projeto 1

Recomendação de alojamento

No primeiro trabalho foi-nos pedido para elaborar um sistema baseado em conhecimento que recolhesse as preferências do utilizador relativamente a moradias para férias e gerasse com base nessas informações, uma recomendação de um alojamento que mais se adequasse às suas expectativas.

Iniciamos este projeto definindo uma base de dados retirada de alguns sites de compra e venda de casas, num formato id,lista, sendo o id o identificador numérico do alojamento e a lista contendo todos os dados relativos a habitação como local, zona, estação do ano, orçamentos, rankings e tipo, armazenada num ficheiro bd.pl.

bd - Bloco de notas

Ficheiro Editar Formatar Ver Ajuda

```
dados('1',['portugal','setubal','praia','TheSeaSideCocoon','40','25','verao','5','nao','32','apartamento','centro','nao']).
dados('2',['portugal','vila Real','piscina','CasadaOliveira','85','20','verao','4.9','sim','52','casa','norte','nao']).
dados('3',['portugal','portalegre','piscina','CasaChãodeOurém','30','8','verao','4.9','sim','19','casa','centro','sim']).
dados('4',['portugal','faro','praia','PensãoBicuar','90','40','verao','4.8','sim','4.8','apartamento','sul','sim']).
dados('5',['portugal','viana do castelo','piscina','CasadoReborido','18','60','inverno','4.9','nao','39','casa','norte','sim']).
dados('6',['portugal','leiria','piscina','CasadosMangues','60','35','primavera','4.9','sim','47','casa','centro','nao']).
dados('7',['portugal','lisboa','piscina','CastelloPrimeSuites','119','56','outono','4.8','sim','87','hotel','centro','sim']).
```

De seguida elaboramos as regras em prolog onde são feitas as queries à bd que com auxílio dos ficheiro forward.pl e proof.pl, código fornecido nas aulas tp, definem a lógica do programa e cruzam as opções do utilizador com a base de dados acima tirando, caso haja uma correspondência o alojamento indicado ao utilizador. [interface1.pl]

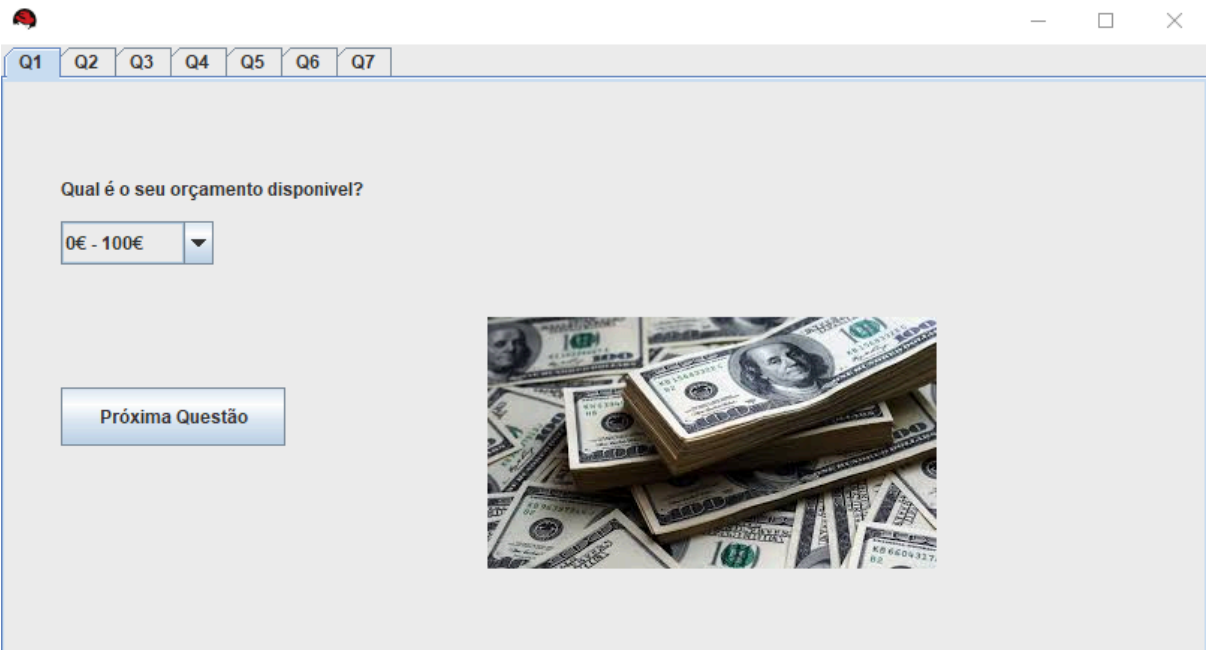
interface1 - Bloco de notas

Ficheiro Editar Formatar Ver Ajuda

```
:- dynamic(fact/1), [forward,proof,bd].  
if preco1 and tipo1 and estacao2 and regioa1 and garagem2 and casa2 and diversao2 then '1'.  
if preco1 and tipo2 and estacao2 and regioa2 and garagem1 and casa1 and diversao2 then '2'.  
if preco1 and tipo2 and estacao2 and regioa1 and garagem1 and casa1 and diversao1 then '3'.  
if preco1 and tipo1 and estacao2 and regioa3 and garagem1 and casa2 and diversao1 then '4'.  
if preco1 and tipo2 and estacao4 and regioa1 and garagem1 and casa1 and diversao1 then '5'.
```

Por fim, decidimos fazer uma interface gráfica com recurso ao IDE Netbeans em código java, onde nos inspiramos num projeto semelhante do github <https://github.com/miguelsilvaw/Prolog---Java---Choose-a-festival-according-to-a-knowledge-database>. Adaptamos alguns dos componentes gráficos, botões e checklists para construir um questionário com 7 questões sobre o utilizador que, de forma indireta, permitam ao utilizador introduzir as suas preferências.

A ligação ao prolog foi feita através de uma biblioteca jpl.jar que contém as funções necessárias para ligar o ambiente java às funções do prolog, e os ficheiros prolog foram carregados e lidos com recursos às classes File, InputStream, FileReader e BufferedReader do java.

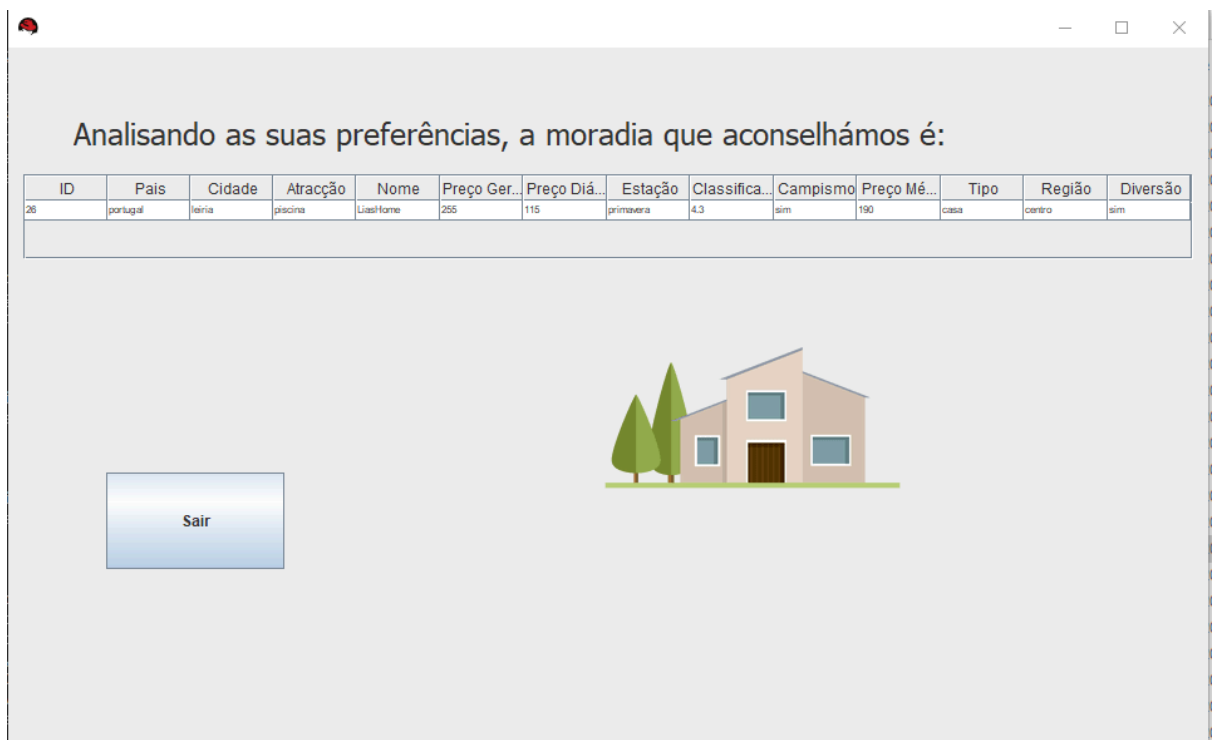


```

public LigacaoProlog(String ficheiroProlog) {
    dados = new ArrayList<>();
    prolog = "consult('" + ficheiroProlog + "')";
    Query prologQuery = new Query(prolog);
    if (prologQuery.hasSolution()) {
        }
    prologQuery.close();
}

```

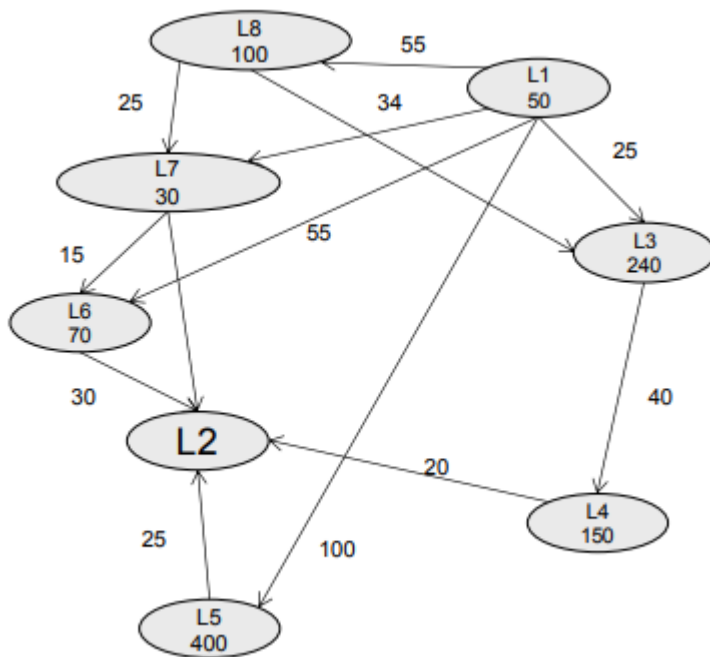
Após a resolução do questionário os dados são adicionados a um ArrayList de Strings com a respostas do utilizador, é feito o consult ao ficheiro prolog e são comparados com os fatos contidos no ficheiro interface1.pl, chamados através da função getSolucao, caso a combinação de respostas dada pelo utilizador não esteja na lista de fatos ele printa a mensagem "Não temos casas para si neste momento.", caso contrário abre uma janela com uma tabela contendo o alojamento encontrado.



Projeto 2

Itinerário

No segundo trabalho tínhamos um grafo contendo um circuito de alojamentos representado pelos pontos com os seus custos, as setas representando os sentidos das direcções com as respectivas distâncias.



Os objetivos deste trabalho foram obter o caminho mais curto e mais barato entre L1 e L2, através de um algoritmo gerar e testar em prolog.

Criamos um ficheiro prolog a partir do código fornecido nos slides 8 e 9 do arquivo da UC que contém os algoritmos de busca dos caminhos e a resolução do caminho mais curto. A lógica deste código é gerar todas as rotas possíveis de forma recursiva depois de definidos os arcos com os pontos de entrada e saída e as respectivas distâncias, e retirar o menor caminho que será o conjunto de arcos visitados com a menor soma de distâncias.

Começamos então por definir os arcos descritos no enunciado e associar os respectivos custos.

```

next(21, 23, 240).
next(21, 25, 100).
next(21, 26, 55).
next(21, 27, 34).
next(21, 28, 55).
next(23, 24, 40).
next(24, 22, 20).
next(25, 22, 25).
next(26, 22, 30).
next(27, 22, 145).
next(27, 26, 15).
next(28, 23, 80).
next(28, 27, 25).
custo(21,50).
custo(22,0).
custo(23,240).
custo(24,150).
custo(25,400).
custo(26,70).
custo(27,30).
custo(28,100).

```

Depois acrescentamos o código do ficheiro caminho.pl adicionando as restrições do enunciado e as variáveis custo e distância como argumentos nas funções; e o do exercício dos arcos que contém as funções que determinam o comprimento alterando o corpo dessa função de modo a que, ao invés de determinar o comprimento, determinar os somatórios dos custo e distância associados a cada ligação, sendo que a distância está associada às arestas e o custos aos pontos (de entrada).

```

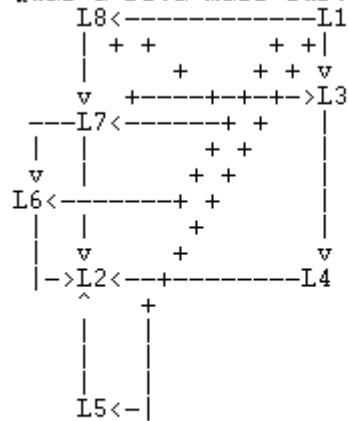
caminhocurto(X,Y,C,P):-caminho(X,Y,C,P), % gerar
    maiscurto(X,Y,C,P).% testar
maiscurto(X,Y,C,P):- comprimento(C,NC,CS),!,
    \+ menor(X,Y,NC,P,CST1),write('Caminho: '),reverse(C,CR),write(CR),write(' Distancia: '),write(NC),write(' Custo: '),write(CS).
menor(X,Y,NC,P,CST):- caminho(X,Y,C1,P),
    comprimento(C1,NC1,CS),
    NC1<NC,CS<450.

% calcula o comprimento de um dado caminho: comprimento(C,N)
comprimento([],0,0).
comprimento([R],0).
comprimento([L|R],N,CS):- comprimento(R,N1,CS1),dst(L,R,MC,CST1),N is N1+MC,CS is CS1+CST1.
%, N is N1+1
dst(_,[],0,0).
%dst(_,R,0).custo(L,CS1),CS is CS1,
dst(L,[R|_],M,CST):-next(R,L,P),custo(R,CS1),M is P,CST is CS1.

```

Depois de testar este algoritmo, decidimos implementar uma pequena interface para correr na consola do utilizador mostrando um menu inicial com uma representação do diagrama do enunciado com 2 opções de resolução do problema e uma opção de saída. Quando o utilizador clica gerar testar ele abre as opções para que escolha caminho de chegada e partida, daqui ele pega essas duas variáveis como x e y e executa a função caminhocurto, printando de seguida o resultado,ou seja, o caminho mais curto, distância e respetivo custo.

Qual a rota mais curta!



- 1 - Gerar e testar
- 2 - Algo A
- 3 - Sair

1.

De que ponto deseja partir?

- 11: 50\$ [21]
- 12: 0\$ [22]
- 13: 240\$ [23]
- 14: 150\$ [24]
- 15: 400\$ [25]
- 16: 70\$ [26]
- 17: 30\$ [27]
- 18: 100\$ [28]

28.

Para que ponto deseja sair?

- 11: 50\$ [21]
- 12: 0\$ [22]
- 13: 240\$ [23]
- 14: 150\$ [24]
- 15: 400\$ [25]
- 16: 70\$ [26]
- 17: 30\$ [27]
- 18: 100\$ [28]

22.

Caminho: [28,27,26,22] Distancia: 70 Custo: 200

Neste teste de cima escolhemos o algoritmo gerar e testar, selecionado como ponto inicial o l8 e final o l2, o caminho mais curto obtido foi o [l8, l7, l6, l2] com distância 70 e custo 200. Depois de alguns testes aumentando e diminuindo algumas distâncias e variando os custos concluímos que o algoritmo era sólido obtendo sempre o caminho mais curto possível dentro da restrição de orçamento dada.

Conclusão

Apesar das dificuldades na conclusão deste projeto, desde os problemas de configuração no primeiro trabalho na integração do prolog com o java devido a problemas de conflitos de versões e no entendimento do prolog sendo a primeira vez que os elementos do grupo tiveram contacto com esta linguagem, a evolução do trabalho foi crescente com o passar do tempo assim como o desenvolvimento se foi tornando mais fácil embora já numa fase mais avançada, ficando assim o projeto algo incompleto, não cumprindo totalmente os objetivos, e uma sensação que poderia ter sido melhorado.