Sistemas de Representação de Conhecimento e Racíocinio (3º ano de Curso)

Trabalho de Grupo 1

Relatório de Desenvolvimento

Gonçalo Nogueira	Carlos Afonso	João Gomes	Pedro Lima	João Gomes
(A86617)	(A82529)	(A82428)	(A80328)	(A82238)

3 de Maio de 2020

Resumo

Neste relatório será apresentado todo o processo de desenvolvimento do primeiro projeto de grupo no âmbito desta unidade curricular, acompanhado de uma explicação intuitiva e fundamentada, derivada das aulas teóricas assim como da presença do grupo nas aulas práticas.

Este trabalho em específico engloba:

- Demonstração da representação de conhecimento positivo e negativo;
- Demonstração da representação de conhecimento imperfeito através da utilização de valores nulos nos tipos estudados;
- Demonstração da manipulação de invariantes cujo propósito é restringir a inserção e remoção do conhecimento
- Desenvolvimento de um sistema de inferência capaz de implementar mecanismos de raciocínio inerentes a este tipo de sistema.

Conteúdo

Capítulo 1

Introdução

No âmbito da unidade curricular de Sistemas de Representação de Conhecimento e Raciocínio foi proposta a realização de um trabalho prático cujo objetivo é a motivação dos alunos face à programação em lógica que tem sido ensinada através da linguagem de programação *PROLOG*.

Este projeto denota entre outros o Conhecimento Imperfeito e a utilização de valores nulos de modo a desenvolver um sistema de inferência.

É importante mencionar o panorama do conhecimento a introduzir na nossa base de dados:

- adjudicante: #IdAd, Nome, NIF, Morada $\rightsquigarrow$ {V,F,D}
- adjudicatária: #IdAda, Nome, NIF, Morada $\rightsquigarrow$ {V,F,D}
- contrato: #IdAd, #IdAda, TipoDeContrato, TipoDeProcedimento, Descrição, Valor, Prazo, Local, Data $\rightsquigarrow$ {V,F,D}

Todo o conteúdo que será abordado nos capítulos seguintes, os diferentes tipos de conhecimento, é importante mencionar que serão todos consistentes quanto a esta definição prévia de adjudicante, adjudicatária e contrato, demonstrando a sua utilização prática no cenário dado pelos docentes da unidade curricular.

O contexto do problema é, como definido pelo enunciado do mesmo, a realização de um sistema de representação de conhecimento e raciocínio capaz de atuar num universo de discurso na área da contratação pública para a realização de contratos para a prestação de servidos. Esta contextualização será explicada com maior exatidão nos capítulos que se seguem.

Capítulo 2

Preliminares

É necessário que, para qualquer sistema, este tenha a capacidade de armazenar e manipular informação. Para lidar com a representação de Informação Incompleta, existem dois sistemas: **Sistema de Base de Dados** e **Sistema de Representação do Conhecimento**.

2.1 Sistema de Base de Dados

A manipulação de informação num Sistema de Base de Dados é feita tendo por base alguns pressupostos:

- **Pressuposto do Mundo Fechado**: toda a informação que não existe mencionada na base de dados é considerada falsa.
- **Pressuposto dos Nomes Únicos**: duas constantes diferentes (que definam valores atómicos ou objetos) designam, necessariamente, duas entidades diferentes do universo de discurso.
- **Pressuposto do Domínio Fechado**:não existem mais objetos no universo de discurso para alem daqueles designados por constantes na base de dados.

2.2 Sistema de Representação de Conhecimento

Ao contrário da forma como a manipulação da informação é feita num **Sistema de Base de Dados** convencional, num **Sistema de Representação de Conhecimento** não se assume que a única informação válida é a representada no sistema e as entidades representadas são as únicas existentes no mundo real. Esta forma de lidar com a informação baseia-se nos seguintes pressupostos:

- **Pressuposto do Mundo Aberto**: podem existir outros factos ou conclusões verdadeiras para alem daqueles representados na base de conhecimento.
- **Pressuposto do Mundo Fechado**: duas constantes diferentes (que definam valores atómicos ou objetos) designam, necessariamente, duas entidades diferentes do universo de discurso.
- **Pressuposto do Domínio Aberto**: podem existir mais objetos do universo de discurso para além daqueles designados pelas constantes da base de conhecimento.

Este trabalho será realizado tendo em conta os 3 pressupostos do sistema de **Representação de Conhecimento** anunciados acima.

Em lógica tradicional, as questões colocadas apenas podem ter 2 respostas possíveis: **Verdadeiro** ou **Falso**. Este facto representa um obstáculo, criando a necessidade de estender o programa de forma a que para cada questão existam 3 respostas possíveis: **Verdadeiro**, **Falso** ou então **Desconhecido**. Esta última resposta acontece, quando não existe informação suficiente que permita concluir que a questão seja Verdadeira ou Falsa.

Capítulo 3

Descrição do Trabalho

Este trabalho tem como objetivo criar uma base de conhecimento e estende-lo de forma a caracterizar um universo de discurso na área da contratação pública para a realização contratos para a prestação de serviços.

Como já foi referido em pontos anteriores, iremos representar conhecimento positivo e negativo, imperfeito recorrendo ao uso de valores nulos, manipular invariantes que representem restrições.

3.1 Base do Conhecimento

Começamos por identificar as diferentes entidades e atributos que nos permitirão caracterizar o universo de discurso na área de contratação pública para a realização de contratos para a prestção de serviços.

- **adjudicante**: caracterizado pelos atributos: *IdAda, Nome, Nif, Morada -> V,F,D.*
- **adjudicataria**: caracterizado pelos atributos: *IdAda, Nome, Nif, Morada -> V,F,D.*
- **contrato**: caracterizado pelos atributos: *IdAd, IdAda, TipoDeContrato, TipoDeProcedimento, Descrição, Valor, Prazo, Local, Data -> V,F,D.*

3.2 Representação do Conhecimento Perfeito

3.2.1 Conhecimento Positivo

O conhecimento positivo relativo aos adjudicantes pode ser visto na figura 3.1.

```
% adjudicante: #IdAd, Nome, Nif, Morada ~{V,F,D}
adjudicante(1, Municipio_de_Braga, 506901173, 'Braga').
adjudicante(2, Teatro_Circo_de_Braga, 500463964, 'Braga').
adjudicante(3, Universidade_do_Minho, 502011378, 'Braga').
adjudicante(4, Centro_Hospitalar_do_Alto_Ave, 508080827, 'Guimarães').
adjudicante(5, Tribunal_Constitucional, 600014193, 'Lisboa').
adjudicante(7, Direcao_Geral_da_Saúde, 600037100, 'Lisboa').
adjudicante(8, Universidade_do_Porto,  501413197, 'Porto').
adjudicante(9, Instituto_Nacional_de_Estatistica, 502237490, 'Lisboa').
adjudicante(10, Municipio_de_Fafe, 506841561, 'Fafe').
```

Figura 3.1: Conhecimento Positivo Adjudicantes

O conhecimento positivo relativo à adjudicataria pode ser visto na figura 3.2.

```
% adjudicataria: #IdAda, Nome, Nif, Morada~{V,F,D}
adjudicataria(1, Soserfis_Contabilidade_Fiscalidade_Gestao_Empresarial_Lda,
    502021209, 'Braga').
adjudicataria(2, Bragaconta_Gestao_Empresarial_Lda, 509689930, 'Braga').
adjudicataria(3, Fafware_Informatica_Unipessoal_Lda, 509861415, 'Fafe').
adjudicataria(4, Sempreluz_Canalizacoes_E_Electricidade_Lda, 500313628,
    'Coimbra').
adjudicataria(5, Pausa_Simpatica_Lda, 514226854, 'Matosinhos').
adjudicataria(6, Enigmalecrim_Lda, 515912794, 'Faro').
adjudicataria(7, Atrevomeasorte_Lda, 513899545, 'Braga').
adjudicataria(8, Everything_Is_New_Lda, 507903480, 'Lisboa').
adjudicataria(9, Diligente_Plano_Unipessoal_Lda, 515785113, 'Aveiro').
adjudicataria(10, Faftir_Transportes_Lda, 505404737, 'Fafe').
```

Figura 3.2: Conhecimento Positivo Adjudicataria

O conhecimento positivo relativo aos contratos pode ser visto na figura 3.3.

```
%contrato: #IdAd, #IdAda, TipoDeContrato, TipoDeProcedimento, Descrição,
    Valor, Prazo, Local, Data~{V,F,D}
contrato(1,2, 'Aquisição de serviços', 'Consulta Previa', 'Consultoria',
    20000, 100, 'Braga', (24,3,2020)).
contrato(3,1, 'Aquisição de serviços', 'Ajuste Direto', 'Assessoria
    jurídica', 1999, 90, 'Braga', (30,3,2020)).
contrato(10,10, 'Aquisição de serviços', 'Concurso Publico', 'Transporte de
    mercadoria', 1500, 60, 'Fafe', (24,4,2020)).
contrato(4,3, 'Aquisição de bens', 'Ajuste direto', 'Compra de material
    informatico', 4000, 30, 'Guimarães', (21,2,2020)).
```

Figura 3.3: Conhecimento Positivo Contrato

3.2.2 Conhecimento Negativo

Ao estender a programação lógica, passamos a poder representar informação negativa explicitamente.

A representação de informação negativa pode ser feita de duas maneiras distintas: recorrendo à **Negação por Falha**, representada normalmente pelo predicado *nao* ou recorrendo à **Negação Forte**, representada normalmente pelo conector -. Fizemo-lo usando **Negação forte**. Podemos ver na figura 3.4.

```
% nao pode ser adjudicante se for adjudicataria
% adjudicante : N -> {V,F,D}
-adjudicante(_, Nome, Nif, Morada) :- adjudicataria(ID, Nome, Nif, Morada).



% adjudicataria : N -> {V,F,D}
-adjudicataria(_, Nome, Nif, Morada) :- adjudicante(ID, Nome, Nif, Morada).


% É falsa a informação de que certos adjudicantes/adjudicatarias tenham um
    contrato.
-adjudicante(11, Municipio_de_Guimaraes, 505948605 , 'Guimarães').
-adjudicante(12, Universidade_de_Lisboa, 510739024 , 'Lisboa').
-adjudicante(13, Universidade_de_Aveiro, 501461108 , 'Aveiro').


-adjudicataria(11, Kyaia_Solucoes_Informaticas_Lda, 509934870, 'Guimarães').
-adjudicataria(12, Ipj_Texteis_Lda, 502752394, 'Braga').
-adjudicataria(13, Convenient_Flavour_Lda, 514418230, 'Braga').


%nao podem existir contratos neste dia por ser feriado.
-contrato(_, _, _, _, _, _, _, _, (1,1,2020)).
```

Figura 3.4: Conhecimento Negativo

3.3 Conhecimento Imperfeito

Face à falta de informação, ou apenas ao acesso a uma parte parcial desta, faz-se uso de **valores nulos** como solução para os casos em que a resposta a uma dada questão será *Desconhecida*.

Usamos 3 tipos distintos de valores nulos:

- **Conhecimento Incerto**: permite representar valores desconhecidos, de entre um número indeterminado de hipóteses.
- **Conhecimento Impreciso**: permite representar valores desconhecidos, de entre um número determinado de hipóteses.
- **Conhecimento Interdito**: permite representar valores desconhecidos e que não são permitidos de conhecer.

3.3.1 Conhecimento Incerto

Neste tipo de valor nulo, não existem quaisquer provas que suportem um dado conhecimento ser verdadeiro ou falso.

Criamos várias excepções que permitem que um adjudicante dê entrada com NIF desconhecido ou morada desconhecida. Também criamos as mesmas excepções para a adjudicatária. No que toca ao contrato, criamos excepções que permitam que um contrato tenha valor desconhecido ou prazo desconhecido.

Podemos ver na figura 3.5, alguns exemplos da implementação de Conhecimento Incerto.

```
% Deu entrada uma Entidade adjudicante com nif desconhecido
adjudicante(14, Municipio de Bragança, incNIF1, 'Bragança').
excecao(adjudicante(IdAd, Nome, NIF, Morada)) :- adjudicante(IdAd, Nome, incNIF1, Morada).
incNIF(incNIF1).

% Deu entrada uma Entidade adjudicante com morada desconhecida
adjudicante(15, Instituto do Emprego e da Formação Profissional, I. P., 501442600, incMorada1).
excecao(adjudicante(IdAd, Nome, NIF, Morada)) :- adjudicante(IdAd, Nome, NIF, incMorada1).
incMorada(incMorada1).


% Adoção do pressuposto do domínio fechado -> adjudicatária
-adjudicatária(IdAda, Nome, NIF, Morada) :- nao(adjudicatária(IdAda, Nome, NIF, Morada)),
                                            nao(excecao(adjudicatária(IdAda, Nome, NIF, Morada))).

% Deu entrada uma Entidade adjudicatária com nif desconhecido
adjudicatária(14, Topgim Material Desportivo e Lazer, Lda., incNIF1, 'Sintra').
excecao(adjudicatária(IdAda, Nome, NIF, Morada)) :- adjudicatária(IdAda, Nome, incNIF1, Morada).
incNIF(incNIF1).

% Deu entrada uma Entidade adjudicatária com morada desconhecida
adjudicatária(15, Otis elevadores, Lda., 500069824, incMorada1).
excecao(adjudicatária(IdAd, Nome, NIF, Morada)) :- adjudicatária(IdAd, Nome, NIF, incMorada1).
incMorada(incMorada1).
```

Figura 3.5: Conhecimento Incerto

3.3.2 Conhecimento Impreciso

Neste tipo de valor nulo, temos um valor finito de hipóteses e sabemos que a opção verdadeira se encontra de entre esses valores, mas não sabemos qual é. Criamos uma excepção em relação à entidade Contrato que permite que um contrato com valor desconhecido seja válido, desde que esse valor esteja entre um intervalo de valores definido. Criamos também uma excepção relativa à Entidade adjudicatária como podemos ver na figura 3.6.

```
% O valor do contrato X encontra-se entre 5000 e 6000 euros
excecao(contrato(3, 9, 'Aquisição de serviços', 'Ajuste Direto Regime Geral', 'Assessoria
    jurídica', X, '500 dias', 'Braga', (21-01-2020))) :- X > 5000, X < 6000.
imprecisoCusto(3,9,(21-01-2020)).

% O valor da morada X pode ser Faro ou Portimao
excecao(adjudicataria(13, Papelaria_Gomes_Lda, 526355224, 'Portimao')).
excecao(adjudicataria(13, Papelaria_Gomes_Lda, 526355224, 'Faro')).
```

Figura 3.6: Conhecimento Impreciso

3.3.3 Conhecimento Interdito

Conhecimento Interdito carateriza um tipo de dados que não é admissível ocorrer na base de conhecimento, assim como um tipo de dados desconhecido.

Aqui, o valor nulo identifica um valor desconhecido assim como representa um valor que não é permitido conhecer. Consequentemente, qualquer tentativa de conhecer este valor será rejeitada, sendo um *request* inconsistente tendo em conta a nossa base de conhecimento.

```
% Por questões de confidencialidade nao se pode saber:
%       - valor
nuloInterditoVal(nuloValor1).
% Impossibilidade de saber o valor correspondente a um determinado contrato.
excecao(contrato(IdAd, IdAda, TipoDeContrato, TipoDeProcedimento, Descricao, Valor, Prazo, Local, Data)) :-
    contrato(IdAd, IdAda, TipoDeContrato, TipoDeProcedimento, Descricao, nuloValor1, Prazo, Local, Data).
contrato(16, 16, 'Empreitadas de obras publicas',
            'Ajuste Direto Regime Geral',
            'Reposiçao do pavimento de valas da rede de agua na EN 207-1, em Barrosas St.º Estêvao', nuloValor1, '20 dias', 'Lousada', (15-03-2020)).
% Invariante para impedir a evoluçao do contrato anterior com interdiçao no valor.
+contrato(Id,Id2,TC,TP,D,V,P,L,DT) ::
    (findall( (Id,Id2,TC,TP,D,V,P,L,DT),
    (contrato(16, 16, 'Empreitadas de obras publicas', 'Ajuste Direto Regime Geral',
            'Reposiçao do pavimento de valas da rede de agua na EN 207-1,
            em Barrosas St.º Estêvao', Va, '20 dias', 'Lousada', (15-03-2020))
    ,nao(nuloInterditoVal(Va))), S),
    comprimento(S,N),
            N==0).
```

Figura 3.7: Conhecimento Interdito

3.4 Manipulação de Invariantes

Para que a base de dados funcione devidamente, é necessária a implementação de invariantes cujo propósito é controlar a inserção e remoção de conhecimento na nossa base.

Os invariantes podem ser divididos em duas classes: Invariante Estrutural e Invariante Referencial. Como o nome indica, o invariante estrutural permite manter a congruência da base de conhecimento, impedindo por exemplo a adição de dois adjudicantes com o mesmo ID:

Já o invariante referencial, restringe os dados na inserção ou remoção de dados à base. Um exemplo disto será por exemplo a confirmação que um NIF é válido:

```
+adjudicante( IdAd, _, _, _) :: ( integer(IdAd),
                                  findall(IdAd, adjudicante(IdAd, Nome, Nif, Morada), S),
                                  comprimento( S, N ), N == 1 ).
```

Figura 3.8: Invariante Estrutural

```
+adjudicante( _, _, Nif, _) :: ( integer(Nif),
                                 Nif >= 100000000, Nif =< 999999999).
```

Figura 3.9: Invariante Referencial

Estes invariantes permitem, como mencionando anteriormente, que a base de conhecimento se mantenha consistente e que funcione corretamente. Isto implica manipular a informação que é inserida, assim como a manipulação de informação que é removida:

```
% Invariante Estrutural: Só se pode eliminar se existir o ID existir na Base de Conhecimento.
-adjudicante( IdAd, _, _, _) :: ( integer(IdAd),
                                  findall(IdAd, adjudicante(IdAd, Nome, Nif, Morada), S),
                                  comprimento( S, N ), N == 0 ).

% Invariante Referencial: Só se pode eliminar adjudicante se nao existirem contratos com esta entidade.
-adjudicante( IdAd, _, _, _) :: ( findall( IdAd, contrato(IdAd, IdAda, TipoDeContrato, TipoDeProcedimento, Descricao, Valor, Prazo, Local, Data), L),
                                  comprimento( L, N),
                                  N == 0 ).

% Invariante Estrutural: Só se pode eliminar se o ID existir na Base de Conhecimento.
-adjudicataria( IdAda, _, _, _) :: ( integer(IdAda),
                                     findall(IdAda, adjudicataria(IdAda, Nome, Nif, Morada), S),
                                     comprimento( S, N ), N == 0 ).

% Invariante Referencial: Só se pode eliminar adjudicatarias se nao existirem contratos com esta entidade.
-adjudicataria( IdAda, _, _, _) :: ( findall( IdAda, contrato(IdAd, IdAda, TipoDeContrato, TipoDeProcedimento, Descricao, Valor, Prazo, Local, Data), L),
                                     comprimento( L, N),
                                     N == 0 ).
```

Figura 3.10: Invariantes

3.5 Problemática da evolução do conhecimento

Este tema está diretamente associado à necessidade de manter a base consistente após a inserção ou remoção de conhecimento na mesma. De modo a assegurar que isto acontece é necessário tomar algumas precauções quanto aos métodos mencionados anteriormente. É necessário garantir que a base de dados se mantém fidedigna após inserções ou remoções de dados.

Isto é conseguido através de um teste, executado na base de conhecimento, que verifica se a alteração efetuado corromperá ou não a base. Isto é conseguido através dos Invariantes do capítulo anterior, assim como nos predicados de inserção e remoção da nossa base de conhecimento.

```
% insercao: T -> {V,F}
insercao(T) :- assert(T).
insercao(T) :- retract(T), !, fail.

% remocao: T -> {V,F}
remocao(T) :- retract(T).
remocao(T) :- assert(T), !, fail.

% teste: L -> {V,F}
teste( [] ).
teste( [I|Is] ) :- I, teste(Is).

% evolucao: T -> {V,F}
evolucao(T) :- findall(I,+T::I,Li),
               insercao(T),
               teste(Li).

% involucao: T -> {V,F}
involucao(T) :- T,
                findall(I,-T::I,Li),
                remocao(T),
                teste(Li).
```

Figura 3.11: Problemática da Evolução

3.6 Sistema de Inferência

Outro atributo essencial deste projeto é a necessidade de desenvolver um sistema de inferência que implemente mecanismos de raciocínio intrínsecos a bases de conhecimento. Este sistema tem como obrigação funcionar como um interpretador de questões capaz de responder a qualquer pergunta colocada pelo utilizador.

A resposta será dada como **Verdadeira**, **Falsa** ou **Desconhecido** - caso as respostas estejam respetivamente presentes na base de conhecimento, negadas na base de conhecimento ou desconhecidas caso não exista informação acerca da mesma na base de conhecimento (isto poderá acontecer caso a resposta não seja verdadeira mas consequentemente não seja negada).

```
%-----------------------------------------------------------------
% Extensao do meta-predicado demo: Questao,Resposta -> {V,F}
demo( Questao,verdadeiro ) :-
    Questao.
demo( Questao,falso ) :-
    -Questao.
demo( Questao,desconhecido ) :-
    nao( Questao ),
    nao( -Questao ).

%-----------------------------------------------------------------
% Extensao do meta-predicado nao: Questao -> {V,F}
nao( Questao ) :-
    Questao, !, fail.
nao( Questao ).
```

Figura 3.12: Sistema de Inferência

Capítulo 4

Conclusões e Sugestões

Através das aulas lecionados ao longo de toda a unidade curricular, nomeadamente as práticas que forneceram um apoio muitíssimo crucial ao desenvolvimento de todo este projeto, o grupo conseguiu efetivamente criar uma base de conhecimento funcional e consistente, demonstrando o seu conhecimento face diferentes tipos de conhecimento, em particular o conhecimento imperfeito, que foi um ponto vital deste projeto.

Aquando o final deste projeto, numa fase fundamentalmente infeliz para a humanidade, que causou uma interrupção ao funcionamento das aulas e disrupção do seu funcionamento natural, temos em conta que o grupo absorveu uma maior parte do plano letivo lecionado até à data.

Surgiram ao longo do projeto erros que o grupo teve mais ou menos dificuldade a superar, mas deparámo-nos com um trabalho final que consideramos satisfatório e essencial para a aprendizagem da unidade curricular.

Quanto a sugestões face o trabalho apresentado, não temos nenhuma a colocar, visto que o acompanhamento das aulas permitiu ao grupo facilmente desenvolver uma base de conhecimento com os requerimentos colocados pelos docentes.

Em suma, o grupo consolidou os conhecimentos adquiridos ao longo do plano curricular de Sistemas de Representação de Conhecimento e Raciocínio, estando agora prontos para numa próxima fase desenvolver um projeto que supere com ainda mais facilidade os problemas que surgiram ao longo deste.