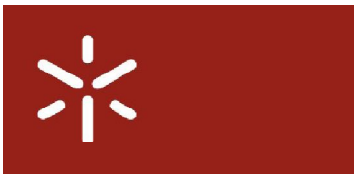

Universidade do Minho

Relatório

Parte 2: Extensão à Programação em Lógica e Conhecimento Imperfeito

Engenharia Biomédica: Ramo de Informática Médica

Representação do Conhecimento

1º Semestre

Ano Lectivo 2009/2010

Autores:

N º 46574 João Vieira

N º 50016 Paulo Marques

N º 50003 Tiago Oliveira

Braga, 4 de Dezembro de 2009

Resumo

No âmbito da Unidade Curricular de Representação do Conhecimento foi realizado um sistema de representação de conhecimento e raciocínio capaz de caracterizar um universo de discurso de forma a descrever um contexto farmacológico. Pretendia-se para tal que se utilizasse a extensão à programação em lógica e que fossem construídos mecanismos que permitissem a representação de conhecimento imperfeito e utilização de valores nulos.

Este sistema é caracterizado por factos certos, por acontecimentos que acarretam dúvidas bem como acontecimentos sobre os quais nunca se poderá ter mais informação do que aquela que se tem. O sistema permite também a evolução do conhecimento, possibilitando ao utilizador acrescentar informação relevante acerca dos medicamentos e suas caixas, tendo para isso que obedecer a determinadas regras fornecidas por invariantes.

O sistema em questão foi então testado através de um caso prático acerca de uma farmácia de casa que simula uma situação real e que pôs à prova as suas potencialidades.

Por fim, verificou-se que o Prolog pode ser utilizado sem recorrer ao pressuposto que indica que toda a informação que não está na base de conhecimento é falsa, e que se podem implementar sistemas que possibilitem o tratamento de informação incompleta.

Índice

Capítulo 1

1.1. Introdução

O presente relatório constitui a segunda parte do trabalho prático no âmbito da Unidade Curricular de Representação do Conhecimento. A sua finalidade é a utilização da extensão à programação em lógica, através do uso da linguagem de programação em lógica Prolog, no âmbito da representação de conhecimento imperfeito, recorrendo à utilização de valores nulos, e da criação de mecanismos de raciocínio adequados.

O universo de discurso situa-se num contexto farmacológico que serviu de base para a construção de um caso prático de modo a ilustrar a capacidade de representação do conhecimento e para a construção de mecanismos de raciocínio.

1.2. Objectivos

Os objectivos deste trabalho prático são os seguintes:

- Representar conhecimento positivo e negativo;
- Representar casos de conhecimento imperfeito, pela utilização de valores nulos de todos os tipos estudados;
- Manipular invariantes que designem restrições à inserção e à remoção de conhecimento do sistema;
- Lidar com a problemática da evolução do conhecimento, criando os procedimentos adequados;
- Desenvolver um sistema de inferência capaz de implementar os mecanismos de raciocínio inerentes a estes sistemas.

Nos capítulos seguintes serão abordados os conhecimentos necessários para elaborar e interagir com a Base de Conhecimento e serão descritas as soluções encontradas, assim como as funcionalidades que estas implementam.

Capítulo 2

2. Base Teórica

2.1. Extensão à programação em lógica

A programação em lógica tradicional determina a veracidade ou falsidade de questões. Num programa em lógica as respostas às questões são apenas de dois tipos: verdadeiro ou falso, verdadeiro para as questões que se concluem ser verdadeiras e falso para aquelas questões sobre as quais não se pode tirar conclusões. A programação em lógica tradicional apoia-se nos seguintes 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 objectos) designam, necessariamente, duas entidades diferentes do universo de discurso;
- **Pressuposto do Domínio Fechado** – não existem mais objectos no universo de discurso para além daqueles designados por constantes na base de dados.

As linguagens tradicionais de programação em lógica proporcionam uma ferramenta poderosa para a representação do conhecimento, uma vez que a característica não-monótona de negação por falha na prova (representada pelo termo não) torna possível a representação de casos de senso comum que não são disponibilizados pela lógica clássica. No entanto a programação em lógica não permite representar directamente a informação incompleta, pois num sistema de raciocínio com uma Base de Conhecimento que possui este tipo de informação contempla questões de representação e de inferência diferentes das que se podem identificar num sistema não-monótono, função do tipo de respostas que se pretendem obter em cada um dos sistemas. Deste modo foi desenvolvida a extensão à programação em lógica que permite a distinção entre três tipos de conclusão para uma questão: verdadeira ou falsa ou, quando não existe informação que permita inferir uma ou outra das conclusões anteriores, a resposta à questão será desconhecida.

Para além de implementar a negação por falha, a extensão à programação em lógica assenta noutro tipo de negação, denominada negação forte. A negação forte é uma forma de identificar a informação negativa ou falsa e é representada pela conectiva "¬". Os dois tipos de negação são bastante diferentes, como se pode ver no seguinte exemplo:

- **Negação por falha:** atravessar ← não comboio;
- **Negação forte:** atravessar ← ¬comboio.

Na negação por falha o automóvel atravessará a linha férrea se não houver prova de que o comboio vai passar, ao passo que na negação forte o automóvel apenas atravessará a linha se houver uma prova que permita concluir que o comboio não vai passar. De forma a ver aplicado o pressuposto do mundo fechado à extensão à programação em lógica, para que a ausência de informação fosse considerada como a negação da sua existência, é necessário adicionar uma regra que formalize o conceito subjacente ao pressuposto do mundo fechado, expressando que é falso tudo aquilo que não for possível provar como sendo verdadeiro. Pode-se definir a regra para um predicado p da seguinte forma:

- ¬*p*(X) ← não *p*(X)

, em que X pode tomar a forma X1, X2, ..., Xn, sendo n o número de argumentos do predicado *p* [(1)].

2.2. Valores Nulos

Em situações de informação incompleta, os valores mais comuns que poderão surgir designam-se por valores nulos. Existem três tipos de valores nulos:

- **Incerto:** permite representar valores desconhecidos e não necessariamente de um conjunto determinado de valores;
- **Impreciso:** representa valores desconhecidos, mas de um conjunto finito e determinado de valores;
- **Interdito**: representa um valor desconhecido e cujo conhecimento é impossível de aceder, ou seja, é um conteúdo que por qualquer razão é inacessível.

São estes os valores que permitem a representação de informação incompleta ou conhecimento imperfeito [(1)].

Capítulo 3

3.1. Descrição do Trabalho

3.1.1. Descrição do caso prático

O caso prático tem como objectivo a demonstração dos mecanismos de raciocínio desenvolvidos e é relativo ao casal Matias que decidiu fazer uma arrumação na secção de medicamentos lá de casa. Como tal, o Sr. Matias ocupou-se de retirar as caixas de medicamentos, uma a uma, dessa secção enquanto a Sra. Matias fez um registo da informação relevante. Desse registo faziam parte os seguintes dados: nome do medicamento, número identificativo da caixa do medicamento, a dosagem e data de validade da caixa, o princípio activo, o laboratório farmacêutico , a indicação terapêutica e a aplicação clínica do medicamento.

3.1.2. Considerações de Implementação

De seguida enumeram-se os casos com os quais o Sr. e a Sra. Matias se depararam e que correspondem ao conhecimento que é necessário representar:

I. O primeiro medicamento a ser registado foi o Mycospor, cuja caixa possuía o número de identificação 1212, dosagem de 50 mg e validade até 31-03-2011. No folheto encontraram-se ainda as informações de que o princípio activo é o bifonasol, como indicação terapêutica é catalogado de antifúngico concebido pelo laboratório Bayer Farmaceutical e clinicamente é aplicado para micoses cutâneas.
II. A segunda caixa a ser processada, possuía o número 1234 e dosagem de 50 mg, era de Vfend e o prazo de validade era 12-12-2010. O Vfend é designado para indicação terapêutica como um antifúngico, cujo princípio activo é o voriconasol, fabricado pelo laboratório Pfizer Limited e é aplicado clinicamente em aspergilose invasiva.
III. O terceiro medicamento a ser retirado não possuía caixa e no folheto constava a informação de que se tratava de Aulin, um anti-inflamatório aplicado clinicamente

no tratamento de dor aguda e osteoartrose dolorosa. Segundo o folheto o laboratório responsável pela sua produção designa-se por Roche, contudo no local respeitante à informação do princípio activo encontrava-se um rasgão e como tal esta informação é desconhecida.

IV. A caixa seguinte vinha vazia. Como tal sabe-se apenas que tem o número 1212, o medicamento designava-se Enjomin, o prazo de validade indicava 31-01-2011 e a dosagem ou era de 100 mg ou de 200mg.

V. A caixa que o Sr. Matias tirou de seguida era de Ceporexim. Do folheto constava a indicação que a cefolexina é o princípio activo deste medicamento e que se trata de um antibiótico fabricado pelo laboratório Erofarm, sendo aplicado clinicamente no tratamento de sinusite e infecção urinária. Na caixa podia ver-se o número 2222 e a dosagem de 500 mg. Relativamente ao prazo de validade, a caixa encontrava-se danificada, mas ambos tinham a certeza que era superior ao dia 25-02-2010.

VI. Foi retirada também uma caixa do medicamento Magnesona, cujo princípio activo é o pidolato de magnésio. Magnesona é um medicamento cuja indicação terapêutica é a nutrição, é produzido pelos Laboratórios Vitória S.A. e aplicado clinicamente em situações de carência em magnésio. A caixa tinha o número 1111, indicava uma dosagem de 1500mg e o prazo de validade era desconhecido, contudo segundo a Sra. Matias terminava em finais do mês de Novembro de 2010.

VII. De seguida o Sr. Matias retirou uma caixa de Ben-u-ron, um antipirético e analgésico cujo princípio activo é o paracetamol, fabricado pela Neo Framaceutica S.A. e aplicado clinicamente em febres e enxaquecas. Na caixa podia ler-se o número 1111, a dosagem era de 500 mg, no entanto, o prazo de validade estava ilegível, sendo impossível de determinar.

VIII. Seguiu-se uma caixa de Aspirina que tinha o número 3333 e validade até 17-07-2010. Relativamente à dosagem, nenhum dos membros do casal tinha a certeza acerca do valor exacto, apenas sabiam que não era de 250 mg. No folheto encontraram a seguinte informação: o princípio activo é ácido acetilsalicílico, a sua indicação terapêutica é a de antipirético, o laboratório farmacêutico responsável pela sua produção é a Bayer Farmaceutical e aplicação clínica em casos de febre.

IX. O Sr. Matias encontrou também uma caixa de Pepsamar com o número 0000 e validade até 17-11-2011. O Sr. Matias sabia que a dosagem não era de 500 mg e a sua mulher tinha a certeza de que a dosagem era superior a 250 mg e inferior a 700 mg.

X. O medicamento que foi retirado em seguida tinha designação de Aero-om. Dimeticone é o princípio activo do medicamento que é fabricado pela Om Pharma S.A. e é indicado para o tratamento clínico de aerofagias. O Sr. Matias afirma ser impossível vir a saber qual a indicação terapêutica deste fármaco.

XI. A próxima caixa possuía o número 9876, tinha validade até 02-01-2011 e era de Pantomicina de 250 mg. No folheto indicava que o princípio activo é a eritromicina e o laboratório que concebeu o medicamento é a Denver Farma. No entanto a Sra. Matias não percebeu qual era indicação terapêutica e nem sabe se deveria ser aplicado para amigdalites ou otites.

XII. O casal Matias afirma que não pode haver duas caixas de medicamento com o mesmo número de identificação.

XIII. Eles chegaram à conclusão que seria um desperdício ter mais de três caixas do mesmo medicamento e portanto não pretendem que esta situação se verifique.

XIV. A Sra. Matias sabe que um dado medicamento contém sempre o mesmo princípio activo, laboratório, indicação terapêutica bem como aplicação clínica. O memo medicamento apenas pode apresentar diferentes dosagens e datas de validade e tem, obrigatoriamente, de apresentar um número de identificação diferente.

3.2. Representação do Conhecimento e Mecanismos de Raciocínio

Predicados *medicamento* e *caixa*

De forma a representar o conhecimento contido nas afirmações do ponto 3.1.2. criaram-se dois predicados: *medicamento* e *caixa*.

O predicado *medicamento* é usado para identificar a o nome do medicamento, o seu princípio activo, a indicação terapêutica, o laboratório farmacêutico responsável pela sua produção e a sua aplicação clínica. Este predicado pode ser representado da seguinte forma:

Extensão do predicado *medicamento:* NC,PA,IT,LAB,AC -> {V,F,D}

Este predicado é constituído por cinco argumentos, sendo o primeiro argumento NC o nome do medicamento, o segundo argumento PA representa o princípio activo, o terceiro argumento IT representa a indicação terapêutica ou categoria em que o medicamento se insere, o terceiro argumento LAB corresponde ao laboratório farmacêutico e por fim o quinto argumento AC representa a aplicação clínica do medicamento. O contradomínio deste predicado é verdadeiro (V), falso (F) e desconhecido (D).

O predicado *caixa* é utilizado para identificar a caixa de um medicamento através do nome do medicamento, o número identificativo da caixa, a dosagem e a data de validade da caixa. O predicado pode ser representado por:

Extensão do predicado *caixa:* NC,NUM,DOS,VAL-> {V,F,D}

O predicado *caixa* é constituído por quatro argumentos. O primeiro argumento NC corresponde ao nome do medicamento, o segundo argumento NUM representa o número identificativo da caixa do medicamento, o terceiro argumento DOS é o valor da dosagem da caixa e por fim o quarto argumento VAL é a data de validade da caixa.

Meta-predicado *demo*

O meta-predicado predicado *demo* faz parte de um sistema de inferência capaz de implementar um mecanismo de raciocínio para a programação em lógica estendida. Este mecanismo suporta três tipos de reposta "verdadeiro", "falso" e "desconhecido". O predicado demo pode ser representado da seguinte forma:

Extensão do meta-predicado *demo:* Questao,Resposta ->{V,F}

O predicado *demo* é constituído por dois argumentos, sendo o primeiro respeitante à questão a colocar ao sistema e o segundo a resposta para a questão colocada. Uma vez que a extensão deste predicado faz parte de um programa em lógica tem como contradomínio verdadeiro, falso. De forma a operar em lógica estendida, adicionam-se as cláusulas do tipo regra que se encontram no Código 1.

Código 1: Cláusulas do meta-predicado *demo*.

```
demo(Questao,verdadeiro) :- Questao, nao(excepcao(Questao)).

demo(Questao,falso) :- -Questao.

demo(Questao,desconhecido) :- nao(Questao), nao(-Questao).

demo(Questao,desconhecido) :- excepcao(Questao).
```

Este predicado é composto por quatro cláusulas do tipo regra. A primeira cláusula indica que a resposta a uma questão (Questão) tem o valor " verdadeiro" se for possível desenvolver uma prova de (Questão) a partir da informação positiva existente na Base de Conhecimento e se esta não for uma excepção. A segunda cláusula define que a resposta a uma questão (Questão) assume o valor "falso" se for possível provar ¬(Questão). A terceira cláusula indica que a resposta a uma questão (Questão) tem o valor "desconhecido" se não existir uma prova de (Questão) nem de ¬(Questão), ou seja se (Questão) não estiver na base do conhecimento assim como ¬(Questão). A quarta cláusula atribui o valor "desconhecido" se (Questão) for uma excepção.

Para a utilização do *demo* foi necessário utilizar a negação forte nos predicados *medicamento* e *caixa* (Código 2).

Código 2: Cláusulas relativas à negação forte dos predicados *medicamento* e *caixa*.

```
-medicamento(NC,PA,IT,LAB,AC) :-
        nao(medicamento(NC,PA,IT,LAB,AC)),
        nao(excepcao(medicamento(NC,PA,IT,LAB,AC))).


-caixa(NC,NUM,DOS,VAL) :-
        nao(caixa(NC,NUM,DOS,VAL)),
        nao(excepcao(caixa(NC,NUM,DOS,VAL))).
```

Com estes predicados assegura-se que se for feita uma questão relativa aos medicamentos e suas caixas resposta vai ser falsa, se não houver informação na Base de

Conhecimento e se não houver nenhuma excepção sobre essa informação.

Meta-predicado *evolucao*

No momento em que surgiu a necessidade de se adicionar informação na base de conhecimento, teve de se ter em conta que esta informação não poderia comprometer a consistência do sistema. Como tal, recorreu-se à construção de um conjunto de invariantes que têm como finalidade prevenir a introdução de informação incoerente. Para se manipular o conhecimento desenvolveu-se o meta-predicado *evolucao*, conforme nos demonstra o Código 3.

Código 3: Meta-predicado *evolucao*.

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

Este predicado permite inserir novo conhecimento no sistema. Nele começa-se por procurar todos os invariantes que foram desenvolvidos para o tipo de termo que se pretende inserir e cada invariante vai ser um elemento da lista. Após a inserção do termo pelo predicado *insercao* é feito um teste à base de conhecimento para verificar se todos os invariantes da lista estão a ser cumpridos, pelo predicado *teste*. Se o facto que se pretende inserir cumprir todas as regras obtidas a partir dos invariantes então esse facto é introduzido. Se o predicado *teste* falhar então é cumprida a segunda cláusula do predicado *insercao* que consiste em retirar o conhecimento inserido.

O predicado *insercao*, representado no Código 4, é constituído por duas cláusulas: a primeira cláusula insere um dado termo na base do conhecimento, já a segunda cláusula retira o termo no caso de ocorrer uma falha no teste (Código 5).

Código 4: Predicado *insercao* auxiliar do meta-predicado *evolucao*.

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

O predicado apresentado no Código 5, permite para uma dada lista, em que cada elemento é uma condição, verificar se todas essas condições são verdadeiras. É composto por duas cláusulas e vai ser usado no meta-predicado *evolucao* de forma a testar se todas as condições impostas pelos invariantes são cumpridas.

Código 5: Predicado *teste* auxiliar do meta-predicado *evolucao*.

```
teste([]).
teste( [X|Y] ) :- X, teste( Y ).
```

Invariantes que restringem os valores a colocar nas datas

De modo a que o cenário descrito se torne o mais próximo possível da realidade, decidiu-se adicionar um conjunto de invariantes, tal como demonstra o Código 6, para as datas de validade dos medicamentos, de tal forma que o dia tem de ser um número entre 1 e 31 (variando conforme os meses), o mês entre 1 e 12 e para o ano não existem quaisquer limitações.

Código 6: Invariantes que permitem manipular datas.

```
+caixa(NC,NUM,DOS,data(Dia,Mes,Ano))::Dia>0.
+caixa(NC,NUM,DOS,data(Dia,Mes,Ano))::(Mes=<12,Mes>0).
+caixa(NC,NUM,DOS,data(Dia,Mes,Ano))::Dia=<28 :- Mes=2.
+caixa(NC,NUM,DOS,data(Dia,Mes,Ano))::Dia=<31 :- pertence(Mes,[1,3,5,7,8,10,12]).
+caixa(NC,NUM,DOS,data(Dia,Mes,Ano))::Dia=<30 :- pertence(Mes,[4,6,9,11]).
```

Estes invariantes permitem definir que o dia e o mês deverão ser representados por números positivos e que o mês deverá ser inferior a 12, além de adequarem os números de dias a cada mês. Deste modo, sempre que se pretenda adicionar informação na base de conhecimento no que diz respeito às datas de validade dos medicamentos, estes invariantes vão garantir que as datas inseridas sejam adequadas o mais próximo possível à realidade. A aplicação destes invariantes pode ser vista na secção A.3 dos Anexos.

Representação do Conhecimento

De seguida apresentam-se as soluções encontradas para representar as afirmações I-IX do ponto 3.1.2.

I.

Neste caso está-se perante informação acerca do medicamento, número de identificação da caixa, dosagem da caixa, data de validade da caixa, princípio activo do medicamento, laboratório e aplicação clínica. Pode ser representada através dos predicados *medicamento* e *caixa* como se demonstra no Código 7.

Código 7: Cláusulas relativas à afirmação I.

```
medicamento(mycospor,bifonasol,antifungico,bayer_farmaceutical,micoses_cutaneas).
caixa(mycospor,1212,50,data(31,03,2011)).
```

Como se está perante um caso de conhecimento perfeito do tipo positivo, ambas as cláusulas constituem factos.

II.

Trata-se novamente de um caso de conhecimento perfeito do tipo positivo, no qual são fornecidas as informações: nome do medicamento, número de identificação da caixa, dosagem da caixa, data de validade da caixa, princípio activo do medicamento, laboratório e aplicação clínica. A representação encontra-se no Código 8.

Código 8: Cláusulas relativas à afirmação II.

```
medicamento(vfend,voriconasol,antifungico,pfizer_limited,apergilose_invasiva).
caixa(vfend,1234,50,data(12,12,2010)).
```

III.

Neste terceiro caso é fornecida informação acerca do nome do medicamento, da indicação terapêutica, da indicação clínica e do laboratório. Contudo desconhece-se o princípio activo. Este conhecimento pode ser representado pelas cláusulas no Código 9.

Código 9: Cláusulas relativas à afirmação III.

```
medicamento(aulin,incerto,anti-inflamatorio,roche,analgesico_e_antipiretico).
excepcao(medicamento(NC,PA,IT,LAB,AC)) :- medicamento(NC,incerto,IT,LAB,AC).
```

O conhecimento relativo ao princípio activo é um caso de conhecimento imperfeito do tipo incerto. De forma a representar este tipo de conhecimento é necessário representar o valor como conhecimento perfeito positivo na base de conhecimento colocando uma palavra reservada no argumento correspondente ao princípio activo, que neste caso á a palavra "incerto", tal como na primeira cláusula. Posteriormente há que representar como excepção as cláusulas que possuem no local do princípio activo a palavra reservada "incerto", como demonstra a segunda cláusula.

Realizando interrogações à Base do Conhecimento através do predicado *demo*, é possível obter o valor "desconhecido" para quando se insere no local do princípio activo um valor qualquer, como evidencia o Código 10.

Código 10: Questão à Base do Conhecimento, usando o predicado *demo*, relativa à afirmação III.

```
|?-demo(medicamento(aulin,nimesulina,antiinflamatorio,roche,analgesico_e_antipiretico),R).
R = desconhecido ? y
yes
```

IV.

O nome do medicamento, o número da caixa e o prazo de validade são as informações que são fornecidas. Contudo desconhece-se o prazo de validade da caixa,

mas sabe-se que este se situa num dado grupo de valores. Esta é uma situação típica de conhecimento imperfeito do tipo impreciso e é representada através das cláusulas do Código 11.

Código 11: Cláusulas relativas à afirmação IV.

```
excepcao(caixa(enjomin,1212,100,data(31,01,2011))).
excepcao(caixa(enjomin,1212,200,data(31,01,2011))).
```

Neste tipo de valor nulo não é possível representar conhecimento perfeito. Constrói-se uma representação explícita das excepções para o conjunto determinado de valores, que neste caso é 100 mg e 200 mg.

Quando se coloca uma questão, usando ao predicado *demo*, à Base de Conhecimento e essa questão possui no argumento da dosagem os valores 100 ou 200 a resposta é "desconhecido". Contudo se for colocado qualquer outro valor a resposta será "falso", tal como se demonstra no conjunto de questões do Código 12.

Código 12: Questões à Base do Conhecimento, usando o predicado *demo*, relativas à afirmação IV.

```
| ?- demo(caixa(enjomin,1212,100,data(31,01,2011)),R).
R = desconhecido ? y
yes
| ?- demo(caixa(enjomin,1212,200,data(31,01,2011)),R).
R = desconhecido ? y
yes
| ?- demo(caixa(enjomin,1212,50,data(31,01,2011)),R).
R = falso ? y
yes
```

V.

Primeiramente, representa-se o conhecimento relativo ao predicado *medicamento* como perfeito positivo, pois todas as informações para os seus argumentos são dadas.

Já para o predicado *caixa*, o caso da data de validade corresponde novamente a um valor nulo do tipo impreciso, no qual a data de validade do medicamento é superior a 25-02-2010. Há então que representar este conhecimento sob a forma de excepção tal como o caso anterior, como se pode verificar no Código 13.

Código 13: Cláusulas relativas à afirmação V bem como o invariante associado às datas.

```
medicamento(ceporexin,cefolexina,antibiotico,eurofarm,sinusite_e_infeccao_urinaria).
excepcao(caixa(ceporexin,2222,500,VAL)) :- dias(VAL,Z),dias(data(25,02,2010),Y),Z>Y.
+caixa(ceporexim,2222,500,VAL)::(dias(VAL,Z),dias(data(25,02,2010),Y),Z>Y).
```

O invariante apresentado no Código 13 é utilizado para fazer evoluir a base de conhecimento, não permitindo inserir datas inferiores ou iguais a 25-02-2010. A demonstração da inserção de conhecimento encontra-se apresentada na secção A.3 em Anexos.

Realizando as questões à base de conhecimento para o predicado *caixa* que se encontram no Código 14, verifica-se que se colocar no argumento correspondente à data de validade uma data anterior ou igual a 25-02-2010, a resposta será “falso”, contudo se a data for posterior à data de referência a resposta será “desconhecido” deixando lugar a que possa ser ou não a data de validade da caixa de Ceporexim.

Código 14: Questões à Base do Conhecimento, usando o predicado *demo*, relativas à afirmação V.

```
| ?- demo(caixa(ceporexim,2222,500,data(29,12,2009)),R).

R = falso ? y

yes

| ?- demo(caixa(ceporexim,2222,500,data(25,02,2010)),R).

R = falso ? y

yes

| ?- demo(caixa(ceporexim,2222,500,data(26,02,2010)),R).

R = desconhecido ? y

yes
```

Para construir a excepção foram utilizados os predicados auxiliares *dias_mes*, *dias e soma* que se encontram na secção A.1 dos Anexos.

VI.

A informação relativa ao medicamento representa-se como conhecimento perfeito.

Esta informação representa um caso de conhecimento imperfeito do tipo impreciso, uma vez que se sabe que o medicamento Magnesona apresenta uma data de validade que expira no final do mês de Novembro de 2010. Contudo, existem várias possibilidades para essa data de validade, tendo-se definido um intervalo de dias (desde o dia 24 até ao 31, inclusive), os quais correspondem ao final do mês. Todos os outros dias não incluídos nesse intervalo são totalmente excluídos.

A representação do conhecimento deve ser feita utilizando uma excepção que englobe todas as possibilidades que não podem ser excluídas. Neste caso, uma vez que estamos perante um elevado conjunto de possibilidades, ter-se-ia que criar uma excepção para cada uma dessas possibilidades, optando-se, por isso, por criar apenas

uma excepção que englobe todas as possibilidades envolvidas no cenário, como se demonstra no Código 15.

Código 15: Cláusulas relativas à afirmação VI.

```
medicamento(magnesona,pidolato_de_magnesio,medicamento_de_nutricao,laboratorios_vitoria_sa
,carencia_de_magnesio).

excepcao(caixa(magnesona,7878,1500,data(DIA,10,2010))) :- DIA>23,DIA=<31.
```

Este é um caso que engloba um valor nulo do tipo desconhecido mas de um conjunto finito e determinado de valores. Esta excepção permite que os valores das datas inseridas não sejam interpretados nem como verdadeiros nem como falsos, mas sim como desconhecidos. Para todos os outros valores de dosagem fora do intervalo incluído na excepção o resultado será "falso".

De seguida, decidiu-se abordar este tipo de conhecimento, colocando questões relativamente ao dia em que expira o prazo de validade do medicamento em causa (Código 16).

Código 16: Questões à Base do Conhecimento, usando o predicado *demo*, relativas à afirmação VI.

```
|?- demo(caixa(magnesona,7878,1500,data(23,10,2010)),R).
R = falso ? y
Yes

| ?- demo(caixa(magnesona,7878,1500,data(30,10,2010)),R).
R = desconhecido ? y
yes
```

Se colocarmos a primeira questão obtém-se uma resposta falsa, pois sabe-se que a data de validade não expira nesse dia, ou seja, o dia colocado na questão encontra-se fora das possibilidades incluídas na excepção. Contudo, quando colocamos a segunda questão, a resposta é desconhecido, pois não é possível desenvolver uma prova para a

questão fazendo uso da primeira cláusula do predicado *demo* por inexistência de informação positiva que a concretize, também não se aplica a segunda cláusula pois existem excepções que impedem que a resposta seja falsa. Assim resta a aplicação da terceira cláusula que determina o valor da resposta como desconhecido.

VII.

A informação relativa ao predicado *medicamento* é representável através de um caso de conhecimento perfeito positivo, como se pode ver no Código 17.

A informação relativa ao predicado caixa representa um caso de conhecimento imperfeito do tipo interdito, uma vez que a data de validade do medicamento Ben-u-ron nunca poderá ser conhecida. Assim, para representar esta informação começou-se por especificar o predicado caixa do medicamento como se pode visualizar no Código 17.

Código 17: Cláusulas relativas à afirmação VII.

```
medicamento(ben-u-ron, paracetamol, antipiretico_e_analgésico, neo_farmaceutica_sa,
febre_e_enxaqueca).

caixa(ben-u-ron,1111,500,interdito).

excepcao(caixa(NC,NUM,DOS,VAL)) :-  caixa(NC,NUM,DOS,interdito).

nulo(interdito).

+caixa(NC,NUM,DOS,VAL)::(solucoes(INT,(caixa(ben-u-ron,1111,500,INT),nao(nulo(INT))),[])).
```

Neste predicado, “interdito” representa um valor nulo que corresponde à data de validade do medicamento em causa. Este valor, para além de identificar valores desconhecidos significa que os valores que representam não são permitidos conhecer, ou seja, no caso de querermos identificar a data de validade do Ben-u-ron, essa tentativa deverá ser rejeitada. Isto porque essa tentativa vai provocar inconsistência na informação constante na Base de Conhecimento. Deste modo, procedeu-se à construção de uma excepção que represente o cenário referido.

Para representar o conhecimento relativo à data foi criado um valor nulo (interdito) e um invariante referencial que rejeita a inserção de qualquer data de validade

no medicamento Ben-u-ron na base de conhecimento, ou seja, o conjunto de soluções para as datas de validade deste medicamento com valor não nulo é um conjunto vazio. Assim, no decorrer da evolução nunca será possível inserir no sistema de representação do conhecimento a data de validade do Ben-u-ron, como se pode ver nas duas últimas linhas do Código 17.

De seguida, decidiu-se abordar este tipo de conhecimento, colocando questões relativamente ao dia em que expira o prazo de validade do medicamento em causa e tentando fazer evoluir o conhecimento através do predicado *evolucao* (Código 18).

Código 18: Questões à Base do Conhecimento, usando o predicado *demo* e *evolucao*, relativas à afirmação VII.

```
| ?- demo(caixa(ben-u-ron,1111,500,data(22,02,2011)),R).

R = desconhecido ? y

Yes

| ?- evolucao(caixa(ben-u-ron,1111,500,data(22,02,2011))).

no
```

Se colocarmos a primeira questão com o predicado *demo* questão obtém-se uma resposta "desconhecido", pois nunca se poderá saber qual a data de validade do Ben-u-ron. A resposta será "desconhecido" para qualquer data que se coloque nesta questão. O sistema também impede a evolução do conhecimento para o valor da data, como se pode ver pela segunda questão, da qual se obtém a resposta "no".

VIII.

Primeiramente a informação relativa ao predicado *medicamento* é claramente um caso de conhecimento perfeito do tipo positivo. No entanto, no caso do predicado *caixa*, não se sabe qual a dosagem de Aspirina pelo que estamos perante conhecimento imperfeito do tipo incerto, pois sabe-se que há uma dosagem específica para o

medicamento, mas não se sabe qual. Assim, representou-se a informação de acordo com o Código 19.

Código 19: Cláusulas relativas aos predicados *medicamento* e *caixa* da afirmação VIII.

```
medicamento(aspirina,acido_acetilsalicilico,antipiretico,bayer_farmaceutical,febre).

caixa(aspirina,3333,naoconhecida,data(17,07,2010)).
```

Neste predicado, "naoconhecida" é um valor nulo do tipo desconhecido que representa a dosagem da Aspirina. A identificação deste valor nulo foi realizada através da introdução de uma excepção no predicado em causa, como se pode visualizar no Código 20.

Código 20: Cláusula relativa à *excepcao* da afirmação VIII.

```
excepcao(caixa(NC,NUM,DOS,VAL)) :- caixa(NC,NUM,naoconhecida,VAL).
```

Esta é uma excepção à concretização da informação negativa, ou seja, quando nos referimos a uma dosagem como sendo a dosagem da Aspirina, não podemos dizer que essa informação é falsa, mas sim desconhecida, dado que na realidade não sabemos qual é o seu valor.

Contudo, sabe-se também que a dosagem da Aspirina não é 250mg, tratando-se este, de um conhecimento perfeito do tipo negativo que é integrado na base de conhecimento recorrendo à negação forte (Código 21).

Código 21: Cláusula relativa à negação forte da afirmação VIII.

```
-caixa(aspirina,3333,250,data(17,07,2010)).
```

Assim, tem-se descrito a informação de que não se sabe qual a dosagem da Aspirina, contudo sabe-se que não é 250mg.

De seguida, decidiu-se abordar este tipo de conhecimento, colocando questões relativamente à dosagem do medicamento em causa (Código 22).

Código 22: Questões à Base do Conhecimento, usando o predicado *demo*, relativas à afirmação VIII.

```
| ?- demo(caixa(aspirina,3333,300,data(17,07,2010)),R).
R = desconhecido ? y
yes
| ?- demo(caixa(aspirina,3333,250,data(17,07,2010)),R).
R = falso ? y
yes
```

Quando se coloca a primeira questão, a reposta dá "desconhecido", precisamente porque o valor da dosagem para a Aspirina é um valor desconhecido para nós. Qualquer valor de dosagem inserido na questão irá dar desconhecido, à excepção de 250mg, como se verifica na segunda questão colocada. Nesta, a resposta é falso, como consequência da negação forte utilizada para aquela dosagem de Aspirina.

IX.

Neste caso, desconhece-se a dosagem do medicamento Pepsamar. Contudo, sabe-se que a dosagem se encontra entre um intervalo de valores, tratando-se por isso, de um caso de conhecimento imperfeito do tipo impreciso. Todas as possibilidades fora desse intervalo são excluídas. A representação deste tipo de conhecimento deve ser feita utilizando a excepção que englobe as possibilidades que não devem ser excluídas (Código 23).

Código 23: Cláusula relativa à *excepcao* da afirmação IX.

```
excepcao(caixa(pepsamar,0000,DOS,data(17,11,2011))) :- DOS>250, DOS<700.
```

Esta excepção permite que os valores de dosagem inseridos não sejam interpretados nem como verdadeiros nem como falsos, mas sim como desconhecidos.

Para todos os outros valores de dosagem fora do intervalo incluído na excepção o resultado será falso. Este é mais um caso que engloba um valor nulo do tipo desconhecido mas de um conjunto finito e determinado de valores.

Além disso, sabe-se ainda que a dosagem da Pepsamar não é 500mg. Trata-se de conhecimento perfeito do tipo negativo e é integrado na base de conhecimento recorrendo à negação forte (Código 24).

Código 24: Cláusula relativa à negação forte da afirmação IX.

```
-caixa(pepsamar,0000,500,data(17,11,2011)).
```

Deste modo, tem-se descrita a informação em que não se sabe qual a dosagem de Pepsamar, sabendo-se, contudo, que não é 500mg.

De seguida, decidiu-se abordar este tipo de conhecimento, colocando questões relativamente à dosagem do medicamento em causa (Código 25).

Código 25: Questões à Base do Conhecimento, usando o predicado *demo*, relativas à afirmação VIII.

```
| ?- demo(caixa(pepsamar,0000,500,data(17,11,2011)),R).

R = falso ? y

yes

| ?- demo(caixa(pepsamar,0000,550,data(17,11,2011)),R).

R = desconhecido ? y

yes

| ?- demo(caixa(pepsamar,0000,800,data(17,11,2011)),R).

R = falso ? y

yes
```

Quando se coloca a primeira questão a resposta dá “falso” devido à negação forte introduzida na base de conhecimento para o medicamento Pepsamar. Quando colocamos a segunda questão a resposta dá “desconhecido” uma vez que o valor de dosagem

inserido se encontra na restrição associada à excepção para o medicamento em causa. Sabemos que 550mg é um possível valor de dosagem, mas não temos garantias. Contudo, quando o valor de dosagem se encontra fora desse intervalo, 800mg por exemplo, a resposta dá falso uma vez que esse valor não constitui uma excepção.

X.

Esta informação representa um caso de conhecimento imperfeito do tipo interdito, visto que a categoria do medicamento Aero-om nunca poderá ser conhecida. Começou-se por especificar o predicado *medicamento* do Aero-om para representar este tipo de conhecimento, através do Código 26.

Código 26: Cláusula relativa ao predicado *medicamento* da afirmação X.

```
medicamento(Aero-om,dimeticone,impossivel,om_pharma_sa,aerofagia).
```

Neste predicado, "impossível" representa um valor nulo que corresponde à categoria do medicamento em causa e que identifica valores desconhecidos. Procedeu-se, então, à construção de uma excepção que não permita conhecer os valores da categoria do medicamento Código 27.

Código 27: Cláusula relativa ao predicado *medicamento* da afirmação X.

```
excepcao(medicamento(NC,PA,IT,LAB,AC)) :- medicamento(NC,PA,impossivel,LAB,AC).
```

Para representar o conhecimento relativo à indicação terapêutica procedeu-se à construção de um valor nulo (impossivel) bem como de um invariante referencial que rejeita a inserção de qualquer categoria no medicamento Aero-om na base de

conhecimento. Deste modo será impossível inserir no sistema de representação do conhecimento indicação terapêutica do Aero-om, no decorrer da evolução (Código 28).

Código 28: Cláusulas relativas ao valor nulo e invariante da afirmação X.

```
nulo(impossível).

+medicamento(NC,PA,IT,LAB,AC) :: solucoes(X,(medicamento(Aero-
om,dimeticone,X,om_pharma_sa,aerofagia),nao(nulo(X))),[]).
```

Sabe-se, ainda, que a indicação terapêutica do Aero-om não é antibiótico, tratando-se este, de um conhecimento perfeito do tipo negativo que é integrado na Base de Conhecimento recorrendo à negação forte (Código 29).

Código 29: Cláusula relativa à negação forte da afirmação X.

```
-medicamento(Aero-om,dimeticone,antibiotico,om_pharma_sa,aerofagia).
```

Tem-se, assim, descrita a informação de que não se sabe qual a categoria de Aero-om e sabe-se, ainda, que não é antibiótico.

Posteriormente, abordou-se este tipo de conhecimento, colocando questões relativamente à indicação terapêutica do medicamento em causa, como se pode ver no Código 30.

Código 30: Questões à Base do Conhecimento, usando os predicados *demo e evolucao*, relativas à afirmação X.

```
| ?- demo(medicamento(Aero-om,dimeticone,antibiotico,om_pharma_sa,aerofagia),R).
R = falso ? y
yes
| ?- demo(medicamento(Aero-om,dimeticone,anti-inflamatorio,om_pharma_sa,aerofagia),R).
R = desconhecido ? y
yes
| ?- evolucao(medicamento(aero-om,dimeticone,anti-pirético,om_pharma_sa,aerofagia)).
no
```

Quando colocada a primeira questão obtém-se a resposta falso uma vez que é sabido que Aero-om não pertence à indicação terapêutica dos antibióticos. Contudo, quando colocamos outra categoria qualquer, tal como anti-inflamatório, a resposta dada é desconhecido, conforme nos demonstra a segunda questão, visto que à excepção de antibiótico, o medicamento Aero-om pode pertencer a qualquer indicação terapêutica. Também quando se tenta introduzir um valor para a indicação terapêutica através do predicado *evolucao*, a resposta que o sistema devolve é "no".

XI.

A informação presente corresponde a conhecimento imperfeito, uma vez que não se sabe se o medicamento Pantomicina era indicado para amigdalites ou otites e qual a indicação terapêutica do mesmo. Está-se, por isso, perante dois tipos de conhecimento imperfeito. O facto de não se saber se o medicamento em causa tem aplicações clínicas para tratar amigdalites ou otites representa conhecimento imperfeito do tipo impreciso. O desconhecimento da sua indicação terapêutica representa conhecimento imperfeito do tipo incerto.

A representação do primeiro tipo de conhecimento deve ser feita com base em excepções, como demonstra o Código 31.

Código 31: Cláusulas relativas ao predicado *excepcao* da afirmação XI.

```
excepcao(medicamento(pantomicina,eritromicina,desconhecida,denver_farma,amigdalite)).
excepcao(medicamento(pantomicina,eritromicina,desconhecida,denver_farma,otite)).
```

Para a representação do segundo tipo de conhecimento recorreu-se, primeiramente, à especificação do predicado medicamento no Código 32.

Código 32: Cláusulas relativas ao predicado *medicamento* da afirmação XI.

```
medicamento(pantomicina,eritromicina,desconhecida,denver_farma,amigdalite).
medicamento(pantomicina,eritromicina,desconhecida,denver_farma,otite).
```

Neste predicado, "desconhecida" é um valor nulo do tipo desconhecido que representa a indicação terapêutica/categoria da Pantomicina. A identificação deste valor nulo do tipo desconhecido foi realizada através da introdução de uma excepção no predicado em causa (Código 33).

Código 33: Cláusula relativa ao predicado *excepcao* da afirmação XI.

```
excepcao(medicamento(NC,PA,IT,LAB,AC)) :- medicamento(NC,PA,desconhecida,LAB,AC).
```

Esta é uma excepção à concretização da informação negativa, ou seja, quando nos referimos a uma categoria como sendo a da Pantomicina, não podemos dizer que essa informação é falsa, mas sim desconhecida, dado que na realidade não sabemos qual é o seu valor.

Seguiu-se uma análise ao conhecimento, colocando questões à categoria e aplicações clínicas do medicamento em causa (Código 34).

Código 34: Questões à Base do Conhecimento, usando o predicado *demo*, relativas à afirmação XI.

```
| ?- demo(medicamento(pantomicina,eritromicina,antibiotico,denver_farma,amigdalite),R).
R = desconhecido ? y
yes
| ?- demo(medicamento(pantomicina,eritromicina,antibiotico,denver_farma,otite),R).
R = desconhecido ? y
yes
| ?- demo(medicamento(pantomicina,eritromicina,antibiotico,denver_farma,febre),R).
R = falso ? y
yes
```

A resposta obtida para a primeira questão é “desconhecido” uma vez que foram inseridos os valores antibiótico para a indicação terapêutica e amigdalite para aplicações clínicas. Isto acontece porque quando se coloca essa questão, antibiótico faz parte das excepções relativas ao predicado medicamento, assim como amigdalite. Ambas constituem excepções, daí a resposta ser ” desconhecido”. O mesmo acontece quando em vez de amigdalite se coloca otite, tal como acontece na segunda questão, uma vez que existe uma excepção que contempla essa aplicação clínica. Para qualquer um dos casos, amigdalite e otite, pode-se inserir qualquer valor para a categoria do medicamento que a resposta dará sempre “desconhecido”. Quando se coloca a terceira questão, ou seja, pergunta-se se existe algum medicamento cuja aplicação clínica seja febre a resposta que nos é dada é “falso”, uma vez que além de não existir informação positiva na base de conhecimento, não existe também qualquer excepção relativa a essa aplicação clínica. Qualquer valor que seja inserido para Pantomicina na categoria, se na aplicação

clínica não se colocar otite ou amigdalite vai ser sempre "falso", visto que não existem excepções que contemplem outras aplicações clínicas que não amigdalite ou otite.

XII.

De forma a garantir que cada caixa apresenta um número único para definir um dado medicamento recorreu-se à construção de um invariante referencial (Código 35).

Código 35: Invariante que garante que cada caixa apresente um único número.

```
+caixa(NC,NUM,DOS,VAL)::(solucoes((NUM),caixa(A,NUM,B,C),S),comprimento(S,N),N=<1).
```

Este invariante procura todas as ocorrências para as caixas com um dado número, que se pretende adicionar, sendo que cada solução vai ser um elemento da lista S. O comprimento dessa lista não pode ser maior do que 1, ou seja, um número NUM não pode representar mais do que uma caixa.

XIII.

A Sra. Matias achava inconveniente ter mais do que 3 caixas de cada medicamento, tendo-se por isso, procedido à construção de um invariante referencial que realizasse essa limitação (Código 36).

Código 36: Invariante que garante que não existam mais de 3 caixas para cada medicamento.

```
+caixa(NC,NUM2,DOS2,VAL2)::(solucoes((NC,NUM,DOS,VAL),caixa(NC,NUM,DOS,VAL),S),comprimento(S,N),N=<3).
```

Este invariante procura todas as ocorrências para as caixas com um dado nome, que se pretende adicionar. Cada solução vai ser um elemento da lista S e o seu comprimento não pode ser maior do que 3, ou seja, um nome NC não pode representar mais do que três caixas.

XIV.

De forma a garantir que não houvesse repetição de conhecimento procedeu-se à construção de um invariante estrutural (Código 37).

Código 37: Invariante que garante que não haja repetição da informação.

```
+medicamento(NC,PA,IT,LAB,AC)::(solucoes((NC,PA,IT,LAB,AC),medicamento(NC,PA,IT,LAB,AC),S),comprimento(S,N),N=<1).
```

Este invariante vai fazer com que não seja permitido inserir conhecimento repetido para um dado medicamento, uma vez que vai procurar todas as ocorrências para os medicamentos com todos os argumentos iguais aos que se pretendem adicionar. Cada solução vai ser um elemento da lista S e o seu comprimento não pode ser maior do que 1, ou seja, não vai haver repetição de conhecimento para um dado medicamento.

Meta-predicado *evolucao1*

Ao inserir nova informação na base de conhecimento, surgiu um problema associado aos casos de conhecimento imperfeito do tipo incerto e de conhecimento imperfeito do tipo impreciso, surgindo a necessidade de se modificar o predicado *evolucao* original, para *evolucao1* (Códigos 38 e 39).

Código 38: Meta-predicado evolucao1 para o predicado *caixa*.

```
evolucao1(caixa(NM,NUM,DOS,VAL)) :-
        excepcao(caixa(NM,NUM,DOS,VAL)),
        retractall(caixa(_,NUM,_,_)),
        retractall(excepcao(caixa(_,NUM,_,_))),
        assert(caixa(NM,NUM,DOS,VAL)),
        solucoes(Invariante,+caixa(NM,NUM,DOS,VAL)::Invariante,Lista),
        teste(Lista).
```

Código 39: Meta-predicado evolucao1 para o predicado *medicamento*.

```
evolucao1(medicamento(NC,PA,IT,LAB,AC)) :-
        excepcao(medicamento(NC,PA,IT,LAB,AC)),
        retractall(medicamento(NC,_,_,_,_)),
        retractall(excepcao(medicamento(NC,_,_,_,_))),
        assert(medicamento(NC,PA,IT,LAB,AC)),
        solucoes(Invariante,+medicamento(NC,PA,IT,LAB,AC)::Invariante,Lista),
        teste(Lista).
```

Através do meta-predicado *evolucao*, era possível inserir esse novo conhecimento, como já aqui foi falado anteriormente. Contudo, as excepções continuavam na base de conhecimento e, por isso, quando se questionava acerca da existência desse conhecimento anteriormente adicionado, a reposta que obtínhamos era "desconhecida", o que não ia de encontro com a realidade. Decidiu-se por isso, implementar um predicado *evolucao1*, utilizado apenas para estes tipos de conhecimento. Este meta-predicado pode apenas ser utilizado no predicado *medicamento* e no predicado *caixa* e apenas em casos em que existam excepções. Em ambos os predicados (medicamento e caixa), ocorre a remoção não só as excepções associadas ao predicado em questão como também, o próprio predicado anterior. Por exemplo, no caso de termos um conhecimento imperfeito do tipo incerto, o predicado que define esse tipo de conhecimento bem como a excepção associada vai ser removida através do comando retractall (para os medicamentos damos como entrada o nome do medicamento, enquanto que para a caixa a entrada corresponde ao seu número identificativo); no caso de termos um conhecimento imperfeito do tipo impreciso a excepção vai ser removida. Deste modo, depois de inserida a nova informação na base de conhecimento, quando se coloca a questão através do predicado *demo*, a resposta vai dar "verdadeiro" e não "desconhecido" como dava anteriormente com o predicado *evolucao*.

De um modo geral, o predicado evolucao1 vai ser utilizado sempre que existam casos de conhecimento imperfeito do tipo incerto ou impreciso, enquanto que para o

resto dos casos se utiliza o predicado *evolucao*. Isto porque também podemos querer adicionar informação sem apagar a que já se encontra na base de conhecimento. Encontram-se algumas demonstrações no Anexo A.4.

Meta-predicado *demo1*

O predicado *demo* até aqui utilizado apenas permitia dar respostas a uma questão. Em determinadas situações torna-se útil saber qual a resposta perante um conjunto de questões. Como tal, procedeu-se à alteração do *demo* como se constata através do Código 40.

Código 40: Meta-predicado *demo1.*

```
demo1([],verdadeiro).

demo1([Questao|Resto],verdadeiro) :- Questao, nao(excepcao(Questao)), demo1(Resto,verdadeiro).

demo1([Questao|Resto],falso) :- -Questao,!.

demo1([Questao|Resto],falso) :- demo1(Resto,falso),!.

demo1([Questao|Resto],desconhecido) :- demo1(Resto,desconhecido),!.

demo1([Questao|Resto],desconhecido) :- nao(Questao), nao(-Questao),!.

demo1([Questao|Resto],desconhecido) :- excepcao(Questao).
```

Quando se tem um conjunto de questões, essas questões são colocadas numa lista. Usando o *demo1*, caso a lista esteja vazia o resultado será verdadeiro (primeira cláusula). Esta cláusula será também utilizada como critério de paragem para uma utilização recursiva do meta-predicado *demo1*. Com este meta-predicado, pretende-se que quando todas as questões sejam verdadeiras, a resposta final seja verdadeira. Isto acontece quando a questão que está à cabeça da lista é verdadeira e todas as questões da cauda também tiveram resultado verdadeiro (segunda cláusula).

A resposta a um conjunto de questões será falso se a questão à cabeça da lista for falsa (terceira cláusula) ou se o resultado das questões da cauda for falsa (quarta cláusula).

Pelas cláusulas anteriores, quando o predicado utilizar a quinta, sexta ou sétimas cláusulas é porque as questões não são todas verdadeiras, nem a primeira é falsa e nem o conjunto das questões da cauda têm resultado falso. Como tal basta tratar os casos em que ou a questão à cabeça é desconhecida (cláusulas seis e sete) com resultado desconhecido, ou o conjunto das questões à cauda apresentam esse mesmo resultado, sendo que a questão à cabeça não é desconhecida (cláusula cinco).

3.3. Conclusões e Sugestões

Com a realização deste trabalho pode-se concluir que os objectivos foram cumpridos, apesar de se poderem ainda implementar algumas melhorias.

O principal aspecto a melhorar consiste na inserção de dados através do predicado *evolucao* e *evolucao1*. Ambos os predicados permitem a inserção de dados quando existe uma negação forte associada aos mesmos, o que não deveria acontecer. Além disso o predicado *evolucao1* está associado aos predicados *medicamento* e *caixa* podendo, por isso, apenas ser utilizado neste contexto farmacológico. Poder-se-ia, por isso, ter recorrido à construção de um *evolucao* geral que pudesse ser invocado para qualquer tipo de contexto. Apesar de algumas tentativas tal não foi possível e, por isso, o predicado *evolucao* final (*evolucao 1*) ficou um pouco limitado, sem no entanto apresentar qualquer influência neste contexto em concreto.

Neste trabalho, deixou-se de parte o pressuposto do mundo fechado, que indica que tudo o que não está na base do conhecimento é falso, tendo-se para isso criado um sistema de inferência capaz de implementar um mecanismo de raciocínio adequado para o cenário em causa.

Pode-se então concluir que este sistema é de elevada importância uma vez que existe informação que, por não se ter a certeza da sua veracidade, não pode ser considerada como verdadeira, mas que também não deverá ser simplesmente ignorada e considerada falsa.

A representação de conhecimento imperfeito também foi um caso de estudo tendo sido representada através de três tipos de valores nulos: incertos, imprecisos e interditos.

Também se desenvolveram mecanismos que visam adicionar informação à base de conhecimento e que ao mesmo tempo garantam consistência na representação do mesmo.

Bibliografia

1. *Representação de informação incompleta.* **Analide, César e José, Neves.** Braga : Universidade do Minho, 1996, Departamento de Informática, pp. 2-29.

Anexos

Anexo A.1: Predicados Auxiliares

Código 41.1.1: Predicado auxiliar *dias_mes*.

```
dias_mes(1,31).
dias_mes(2,28).
dias_mes(3,31).
dias_mes(4,30).
dias_mes(5,31).
dias_mes(6,30).
dias_mes(7,31).
dias_mes(8,31).
dias_mes(9,30).
dias_mes(10,31).
dias_mes(11,30).
dias_mes(12,31).
```

Código 42.1.2: Predicado auxiliar *dias* que converte uma data num total de dias.

```
dias(data(Dia,Mes,Ano),Total):-
        solucoes(Dias,(dias_mes(X,Dias),X<Mes),L),
        soma(L,D),
        Total is D+Dia+Ano*365.
```

Código 43.1.3: Predicado auxiliar *soma*, que soma todos os elementos de uma lista.

```
soma([],0).
soma([A|B],T) :- soma(B,ST), T is A+ST.
```

Anexo A.2: Questões que podem ser colocadas à base de conhecimento

Uma vez que o primeiro e o segundo caso prático representam o mesmo tipo de conhecimento, encontram-se apenas registadas as questões colocadas ao primeiro. Os casos 12, 13 e 14 são casos de invariantes, estando exemplificados na secção A.3.

Código 44.2.1: Questões que podem ser colocadas no primeiro caso prático.

```
| ?- demo(medicamento(mycospor,bifonasol,antifungico,bayer_farmaceutical,micoses_cutaneas),R).
R = verdadeiro ? y
Yes
| ?- demo(medicamento(mycospor,bifonasol,antibiotico,bayer_farmaceutical,micoses_cutaneas),R).
R = falso ? y
yes
| ?- demo(caixa(mycospor,1212,50,data(31,03,2011)),R).
R = verdadeiro ? y
Yes
| ?- demo(caixa(mycospor,1213,50,data(31,03,2011)),R).
R = falso ? y
Yes
```

Código 45.2.2: Questões que podem ser colocadas no terceiro caso prático.

```
| ?- demo(medicamento(aulin,cefolexina,anti-inflamatorio,roche,analgesico_e_antipiretico),R).

R = desconhecido ? y

Yes


| ?- demo(medicamento(aulin,voricosanol,anti-inflamatorio,roche,analgesico_e_antipiretico),R).

R = desconhecido ? y

Yes
```

Código 46.2.3: Questões que podem ser colocadas no quarto caso prático.

```
| ?- demo(caixa(enjomin,1212,100,data(31,01,2011)),R).

R = desconhecido ? y

Yes

| ?- demo(caixa(enjomin,1212,200,data(31,01,2011)),R).

R = desconhecido ? y

Yes

| ?- demo(caixa(enjomin,1212,300,data(31,01,2011)),R).

R = falso ? y

Yes
```

Código 47.2.4: Questões que podem ser colocadas no quinto caso prático.

```
| ?- demo(caixa(ceporexin,2222,500,data(25,02,2010)),R).

R = falso ? y

Yes


| ?- demo(caixa(ceporexin,2222,500,data(26,02,2010)),R).

R = desconhecido ? y

Yes
```

Código 48.2.5: Questões que podem ser colocadas no sexto caso prático.

```
| ?- demo(caixa(magnesona,7878,1500,data(23,10,2010)),R).

R = falso ? y

Yes

| ?- demo(caixa(magnesona,7878,1500,data(31,10,2010)),R).

R = desconhecido ? y

Yes
```

Código 49.2.6: Questões que podem ser colocadas no sétimo caso prático.

```
| ?- demo(caixa(ben-u-ron,1111,500,data(24,06,2011)),R).

R = desconhecido ? y

Yes
```

Código 50.2.7: Questões que podem ser colocadas no oitavo caso prático.

```
| ?- demo(caixa(aspirina,3333,250,data(17,07,2010)),R).

R = falso ? y

Yes

| ?- demo(caixa(aspirina,3333,500,data(17,07,2010)),R).

R = desconhecido ? y

Yes
```

Código 51.2.8: Questões que podem ser colocadas no nono caso prático.

```
| ?- demo(caixa(pepsamar,0000,500,data(17,11,2011)),R).

R = falso ? y

yes

| ?- demo(caixa(pepsamar,0000,269,data(17,11,2011)),R).

R = desconhecido ? y

yes

| ?- demo(caixa(pepsamar,0000,969,data(17,11,2011)),R).

R = falso ? y

Yes
```

Código 52.2.9: Questões que podem ser colocadas no décimo caso prático.

```
| ?- demo(medicamento(aero-om,dimeticone,antibiotico,om_pharma_sa,aerofagia),R).

R = falso ? y

Yes

| ?- demo(medicamento(aero-om,dimeticone,antipiretico,om_pharma_sa,aerofagia),R).

R = desconhecido ? y

Yes
```

Código 53.2.10: Questões que podem ser colocadas no décimo-primeiro caso prático.

```
| ?- demo(medicamento(pantomicina,eritromicina,antibiotico,denver_farma,amigdalite),R).
R = desconhecido ? y
Yes

| ?- demo(medicamento(pantomicina,eritromicina,antibiotico,denver_farma,otite),R).
R = desconhecido ? y
Yes

| ?- demo(medicamento(pantomicina,eritromicina,antibiotico,denver_farma,febre),R).
R = falso ? y
yes
```

Anexo A.3: Demonstração do predicado *evolucao, evolucao1,* bem como algumas questões.

Código 54.3.1: Demonstração do *evolucao1* seguido de uma questão à base de conhecimento para o terceiro caso prático.

```
| ?- evolucao1(medicamento(aulin,propanol,anti-inflamatorio,roche,analgesico_e_antipiretico)).

yes

| ?- demo(medicamento(aulin,propanol,anti-inflamatorio,roche,analgesico_e_antipiretico),R).

R = verdadeiro ? y

yes
```

Código 55.3.2: Demonstração do *evolucao1* seguido de uma questão à base de conhecimento para o quarto caso prático.

```
| ?- evolucao1(caixa(enjomin,1212,300,data(31,01,2011))).

no

| ?- evolucao1(caixa(enjomin,1212,100,data(31,01,2011))).

yes

| ?- demo(caixa(enjomin,1212,100,data(31,01,2011)),R).

R = verdadeiro ? y

yes
```

Código 56.3.3: Demonstração do *evolucao1* seguido de uma questão à base de conhecimento para o quinto caso prático.

```
| ?- evolucao1(caixa(ceporexim,2222,500,data(25,02,2010))).

no

| ?- evolucao1(caixa(ceporexim,2222,500,data(26,02,2010))).

yes

| ?- demo(caixa(ceporexim,2222,500,data(26,02,2010)),R).

R = verdadeiro ? y

yes
```

Código 57.3.4: Demonstração do *evolucao1* seguido de uma questão à base de conhecimento para o sexto caso prático.

```
| ?- evolucao1(caixa(magnesona,7878,1500,data(23,10,2010))).

no

| ?- evolucao1(caixa(magnesona,7878,1500,data(31,10,2010))).

yes

| ?- demo(caixa(magnesona,7878,1500,data(31,10,2010)),R).

R = verdadeiro ? y

yes
```

Código 58.3.5: Demonstração do *evolucao1* seguido de uma questão à base de conhecimento para o sétimo caso prático.

```
| ?- evolucao1(caixa(ben-u-ron,1111,500,data(22,05,2011))).

no

| ?- evolucao1(caixa(ben-u-ron,1111,500,data(11,07,2011))).

no
```

Código 59.3.6: Demonstração do *evolucao1* seguido de uma questão à base de conhecimento para o oitavo caso prático.

```
| ?- evolucao1(caixa(aspirina,3333,500,data(17,07,2010))).

yes

| ?- demo(caixa(aspirina,3333,500,data(17,07,2010)),R).

R = verdadeiro ? y

yes
```

Código 60.3.7: Demonstração do *evolucao1* seguido de uma questão à base de conhecimento para o nono caso prático.

```
| ?- evolucao1(caixa(pepsamar,0000,700,data(17,11,2011))).

no

| ?- evolucao1(caixa(pepsamar,0000,699,data(17,11,2011))).

yes

| ?- demo(caixa(pepsamar,0000,699,data(17,11,2011)),R).

R = verdadeiro ? y

yes
```

Código 61.3.8: Demonstração do *evolucao1* seguido de uma questão à base de conhecimento para o décimo caso prático.

```
| ?- evolucao1(medicamento(aero-om,dimeticone,anti-piretico,om_pharma_sa,aerofagia)).

no
```

Código 62.3.9: Demonstração do *evolucao1* seguido de uma questão à base de conhecimento para o décimo-primeiro caso prático.

```
| ?- evolucao1(medicamento(pantomicina,eritromicina,anti-piretico,denver_farma,febre)).

no

| ?- evolucao1(medicamento(pantomicina,eritromicina,anti-piretico,denver_farma,otite)).

yes

| ?- demo(medicamento(pantomicina,eritromicina,anti-piretico,denver_farma,otite),R).

R = verdadeiro ? y

yes
```

Código 63.3.10: Demonstração do *evolucao1* seguido de uma questão à base de conhecimento para o décimo-primeiro caso prático.

```
| ?- evolucao1(medicamento(pantomicina,eritromicina,anti-piretico,denver_farma,amigdalite)).

yes

| ?- demo(medicamento(pantomicina,eritromicina,anti-piretico,denver_farma,amigdalite),R).

R = verdadeiro ? y

yes
```

Código 64.3.11: Demonstração do invariante, presente no décimo-segundo caso prático, num medicamento do primeiro caso prático.

```
| ?- evolucao(caixa(mycospor,1212,50,data(31,03,2011))).

no
```

Código 65.3.12: Demonstração do invariante, presente no décimo-terceiro caso prático, num medicamento do primeiro caso prático.

```
| ?- evolucao(caixa(mycospor,1213,50,data(31,03,2011))).

yes

| ?- evolucao(caixa(mycospor,1214,50,data(31,03,2011))).

yes

| ?- evolucao(caixa(mycospor,1215,50,data(31,03,2011))).

no

| ?- demo(caixa(mycospor,1213,50,data(31,03,2011)),R).

R = verdadeiro ? y

yes

| ?- demo(caixa(mycospor,1214,50,data(31,03,2011)),R).

R = verdadeiro ? y

yes

| ?- demo(caixa(mycospor,1215,50,data(31,03,2011)),R).

R = falso ? y

yes

| ?- demo(caixa(mycospor,1212,50,data(31,03,2011)),R).

R = verdadeiro ? y

yes
```

Código 66.3.13: Demonstração do invariante, presente no décimo-quarto caso prático, num medicamento do primeiro caso prático.

```
| ?-
evolucao(medicamento(mycospor,bifonasol,antifungico,bayer_farmaceutical,micoses_cutaneas)).

no
```

Anexo A.4: Demonstração do predicado *demo1.*

Código 67.4.1: Demonstração do predicado demo1 para duas questões verdadeiras.

```
| ?-
demo1([medicamento(mycospor,bifonasol,antifungico,bayer_farmaceutical,micoses_cutaneas),medi
camento(vfend,voriconasol,antifungico,pfizer_limited,apergilose_invasiva)],R).

R = verdadeiro ? y

Yes
```

Código 68.4.2: Demonstração do predicado demo1 para duas questões, sendo uma delas verdadeira e outra falsa.

```
| ?-
demo1([medicamento(mycospor,bifonasol,antifungico,bayer_farmaceutical,micoses_cutaneas),medi
camento(vfend,voriconasol,antifungico,pfizer_limited,febre)],R).

R = falso ? y

Yes
```

Código 69.4.3: Demonstração do predicado demo1 para duas questões, sendo uma delas verdadeira e outra desconhecida.

```
| ?-
demo1([medicamento(mycospor,bifonasol,antifungico,bayer_farmaceutical,micoses_cutaneas),medi
camento(aulin,bifonasol,anti-inflamatorio,roche,analgesico_e_antipiretico)],R).

R = desconhecido ? y

Yes
```

Código 70.4.4: Demonstração do predicado demo1 para duas questões, sendo uma delas falsa e outra desconhecida.

```
| ?-
demo1([medicamento(vfend,voriconasol,antifungico,pfizer_limited,febre),medicamento(aulin,bifonasol,anti-inflamatorio,roche,analgesico_e_antipiretico)],R).

R = falso ? y

Yes
```

Código 71.4.5: Demonstração do predicado demo1 para duas questões falsas.

```
|?-
demo1([medicamento(vfend,voriconasol,antifungico,pfizer_limited,febre),medicamento(magnesona,pidolato_de_magnesio,medicamento_de_nutricao,laboratorios_vitoria_sa,febre)],R).

R = falso ? y

Yes
```

Código 72.4.6: Demonstração do predicado demo1 para duas questões desconhecidas.

```
| ?- demo1([medicamento(aulin,bifonasol,anti-inflamatorio,roche,analgesico_e_antipiretico),caixa(pepsamar,0000,600,data(17,11,2011))],R).

R = desconhecido ? y

yes
```