

## Segundo Trabalho de I.A. (Shell de Sistema Especialista) (Prof. Alexandre Direne - 2012/1)

### Atenção:

Este trabalho é obrigatório e deverá ser entregue, impreterivelmente, até o dia 17 de outubro 2012. A solução é individual e deverá ser arquivada no diretório “~alex/IA/” onde o nome do arquivo terá como prefixo o seu nome-de-usuário no sistema do laboratório e, como extensão, “.pl” para indicar que seu conteúdo possui um programa em Prolog. Assim, por exemplo, se o seu nome de usuário no sistema fosse “grs09” então o nome do arquivo seria “grs09.pl” (dentro do diretório “~alex/IA/”). Não se esqueça de proteger completamente o arquivo criado, de maneira a permitir a leitura do mesmo apenas por você! Isso pode ser feito aplicando `chmod og-rwx grs09.pl` antes de efetuar a cópia com a preservação das permissões (`cp -p grs09.pl ~alex/IA/`). Não se preocupe com as permissões do professor que irá corrigir o trabalho. A correção dos trabalhos será parcialmente automatizada, sendo assim, é importante que todos os arquivos com as soluções individuais estejam no diretório citado acima, dentro do prazo estipulado. Não será permitida a entrega do arquivo por e-mail.

### Enunciado:

Implementar um predicado em Prolog, denominado diagnostico, o qual realiza a tarefa de interpretar Regras de Produção. Isso equivale a dizer que tal interpretador é um arcabouço (*Shell*) para a construção de Sistemas Especialistas (*Expert System Shell*). As estratégias de controle do predicado diagnostico devem ser semelhantes às da *Shell* denominada *EMYCIN*, renomada por ser uma das primeiras da história dos Sistemas Especialistas. As simplificações permitidas neste trabalho são as seguintes:

- as fórmulas numéricas são um subconjunto das do *EMYCIN*;
- a interpretação das regras só produz uma conclusão;
- a estrutura gramatical da antecedente de uma regra de inferência admite apenas a interseção lógica entre as premissas que a compõem;
- não há um mecanismo de contexto refinado na interação com o usuário, o que fará com que o programa só consiga conversar sobre um único objeto para determinar o valor de seus atributos.

Para ilustrar uma aplicação do predicado diagnostico, veja o diálogo abaixo:

```
?- consult(grs09).
?- diagnostico(especie).
```

```
Me informe sobre pernas ? 2.
Qual eh o seu grau de exatidao (um numero entre -1 e 1) ? 1.
```

```
Me informe sobre altura ? baixa.
Qual eh o seu grau de exatidao (um numero entre -1 e 1) ? -0.2.
```

```
As hipoteses sao as seguintes: [[humana 1.0]]
```

```
?-
```

Tal diálogo resultou da tentativa de inferir um valor para o atributo especie do objeto que poderia ser uma “instância” de uma das duas classes possíveis: humana ou cão. O código das inferências sobre humanos e cães se encontra todo no único termo do predicado denominado base\_de\_regras, mas que pode ser mudado para representar outro objeto qualquer (*e.g.*, uma instância das classes de doenças infecciosas). Na verdade, as representações do exemplo atual sobre humanos e cães são apenas fragmentos que incluem alguns detalhes combinando atributos com valores qualitativos e quantitativos, tais como a idade (tempo de vida), o sexo, o número de pernas e a altura. A Base de Regras contendo essas representações é a seguinte:

```
base_de_regras(
[
[ [idade, jovem], [especie, humana], [sexo, masculino], -> , 1.0, [chamado, menino] ],
[ [idade, jovem], [especie, humana], [sexo, feminino], -> , 1.0, [chamado, menina] ],
[ [idade, adulto], [especie, humana], [sexo, masculino], -> , 1.0, [chamado, homem] ],
[ [idade, adulto], [especie, humana], [sexo, feminino], -> , 1.0, [chamado, mulher] ],
[ [idade, jovem], [especie, cao], -> , 1.0, [chamado, filhote] ],
[ [idade, adulto], [especie, cao], [sexo, feminino], -> , 1.0, [chamado, cadela] ],
```

```
[ [idade, adulto], [especie, cao], ->, 0.5, [chamado, cao] ],
[ [pernas, 2], ->, 1.0, [especie, humana] ],
[ [pernas, 4], ->, -1.0, [especie, humana] ],
[ [pernas, 4], ->, 0.5, [especie, cao] ],
[ [altura, baixa], ->, 0.5, [especie, cao] ],
[ [altura, alta], ->, 0.5, [especie, humana] ]
]).
```

Note que cada item da Base de Regras é codificado como uma lista dinâmica em Prolog que tem o formato de uma Regra de Produção. Cada uma dessas regras, por sua vez, é composta dos seguintes itens:

- uma sequência de elementos os quais expressam a antecedente da regra como uma interseção lógica (cada elemento é codificado por meio de uma lista dinâmica em Prolog da forma [*<atributo>*, *<valor>*]);
- o sinal de implicação lógica ('->');
- um número (*Real*) que expressa o grau de exatidão (ou *fuzziness* na língua inglesa) da consequente da regra, cujo significado é dado por uma escala de valores entre três limites:
  - 1.0: a antecedente é evidência conclusiva de que a consequente não é verdadeira;
  - 0.0: a antecedente não diz nada sobre o valor-verdade da consequente;
  - 1.0: a antecedente é evidência conclusiva de que a consequente é verdadeira.
- um único elemento que expressa a consequente da regra, também codificado por meio de uma lista dinâmica em Prolog da forma [*<atributo>*, *<valor>*].

Durante um cálculo do Prolog por meio do predicado *diagnostico*, cada pergunta gerada pelo programa irá requerer do usuário tanto o valor de algum atributo como um número sintetizando o *grau de exatidão* que o usuário tem sobre o referido valor (o significado desse número é semelhante ao do número fornecido na consequente de uma regra de inferência).

Para atualizar (ou revisar) os valores de exatidão em cada passo do processo inferencial, o *EMYCIN* calcula a probabilidade resultante a partir da seguinte comparação entre a probabilidade antiga (que está na base de fatos - ver predicado *base\_de\_fatos* abaixo) e a nova (que vem de inferências recursivas assim como de respostas do usuário):

```
Se antiga > 0 e nova > 0 Entao
  resultante = antiga + nova - (antiga * nova);
Se antiga < 0 e nova < 0 Entao
  resultante = antiga + nova + (antiga * nova);
Senao
  (antiga + nova) / (1 - min(abs(antiga),abs(nova)));
Fim-Se
```

Em qualquer ponto do diálogo, o usuário pode perguntar "*porque*" ao invés de fornecer um valor de atributo. Veja o exemplo:

```
?- diagnostico(especie).

Me informe sobre pernas ? porque.
Eu estou tentando usar a regra:
  [[pernas 2] -> 1.0 [especie humana]]

Me informe sobre pernas ? ...
...
```

Os valores da escala de exatidão também são utilizados em mensagens de *monitoramento* e nas conclusões impressas no final do processo de inferência. O monitoramento pode ser ativado ou desativado de acordo com a presença ou não do predicado nulario denominado *monitoramento* no código Prolog da *Shell*. Veja um exemplo de ativação do predicado *diagnostico* com a presença de *monitoramento*.

```
?- consult(grs09).
?- assertz(monitoramento).
?- diagnostico(especie).

>> Descobrimo sobre especie.
```

```
>> Tentando a regra [[pernas 2] -> 1.0 [especie humana]].
>> Descobrimo sobre pernas.
```

Me informe sobre pernas ? 4.

Qual eh o seu grau de exatidao (um numero entre -1 e 1) ? 0.6.

```
>> A exatidao de pernas = 4 passou a ser 0.6.
>> Abandonando esta regra para especie.
>> Tentando a regra [[pernas 4] -> -1.0 [especie humana]].
>> Descobrimo sobre pernas.
>> A exatidao de especie = humana agora eh -0.6.
>> Tentando a regra [[pernas 4] -> 0.5 [especie cao]].
>> Descobrimo sobre pernas.
>> A exatidao de especie = cao agora eh 0.3.
>> Tentando a regra [[altura baixa] -> 0.5 [especie cao]].
>> Descobrimo sobre altura.
```

Me informe sobre altura ? alta.

Qual eh o seu grau de exatidao (um numero entre -1 e 1) ? 0.9.

```
>> A exatidao de altura = alta agora eh 0.9.
>> Abandonando esta regra para especie.
>> Tentando a regra [[altura alta] -> 0.5 [especie humana]].
>> Descobrimo sobre altura.
>> A exatidao de especie = humana agora eh -0.272727.
```

As hipoteses sao as seguintes: [[cao 0.3] [humana -0.272727]]

Observação: assumo que a resposta do usuário sempre fará referência a valores que constam na base de regras.

A *Base de Fatos* (veja a teoria de Sistemas de Produção) é utilizada para armazenar informações sobre os atributos, seus valores e o grau de exatidão sobre cada valor. Cada fato da Base de Fatos tem seu formato dado pelo exemplo abaixo (note que o atributo *altura* armazena informações sobre mais de um valor neste caso):

```
[altura, [baixa, 0.3], [alta, -0.4]]
```

Isso significa uma exatidão de 0.3 sobre a altura da instância ser *baixa* e uma exatidão de  $-0.4$  sobre a altura da instância ser *alta*. O código da Base de Fatos se encontra todo no único termo do predicado denominado *base\_de\_fatos* (veja o exemplo de Base de Fatos depois que termina a inferência do primeiro diálogo, assim):

```
?- base_de_fatos(BF).
```

```
BF = [[altura, [baixa, -0.2]], [especie, [humana, 1.0]], [pernas, [2, 1.0]]] ?
```

```
yes
```

É importante esvaziar completamente a Base de Fatos toda vez que uma nova consulta tiver início. Isso pode ser feito assim no seu programa:

```
diagnostico(Atributo) :-
    retractall(base_de_fatos(_)),
    assertz(base_de_fatos([])),
    ...
```

O predicado unário denominado *limiar\_de\_exatidao* guarda em seu termo um número *L*, também entre  $-1.0$  e  $1.0$ . Tal número é utilizado para fazer com que o processo de diagnóstico *abandone* a avaliação de qualquer regra cujo grau de exatidão da antecedente tenha atingido um valor menor que *L*. Utilize  $L = 0.2$  como um bom valor para este trabalho, tal que:

```
?- limiar_de_exatidao(L).
```

```
L = 0.2 ?
```

```
yes
```

```
?-
```

**Ao estudar o assunto sobre Sistemas de Produção nos livros, tente responder às seguintes perguntas:**

- O que é um Sistema Especialista?
- O quão profundo é o conhecimento do sistema sobre o domínio? Veja como o conhecimento é representado no *EMYCIN* em:  
[http://www.inf.ufpr.br/alex/ARTIGOS\\_IA/Shortliffe\\_et\\_al\\_1975.pdf](http://www.inf.ufpr.br/alex/ARTIGOS_IA/Shortliffe_et_al_1975.pdf)  
Tente analisar também a seguinte visão geral sobre sistemas especialistas da mesma época em:  
[http://www.inf.ufpr.br/alex/ARTIGOS\\_IA/Shortliffe\\_et\\_al\\_1979.pdf](http://www.inf.ufpr.br/alex/ARTIGOS_IA/Shortliffe_et_al_1979.pdf)
- Como o conhecimento está representado?
- Quais são os principais componentes do programa criado? Como exemplo resumido do algoritmo do *EMYCIN*, veja a função *FINDVALUEOF* no seguinte artigo clássico:  
[http://www.inf.ufpr.br/alex/ARTIGOS\\_IA/Davis\\_Buchanan\\_Shortliffe\\_1977.pdf](http://www.inf.ufpr.br/alex/ARTIGOS_IA/Davis_Buchanan_Shortliffe_1977.pdf)
- Como as inferências estão organizadas?
- Quais são os papéis exercidos pelas “*probabilidades*” como valores dos graus de exatidão na representação do conhecimento e no processo de inferência? Como esses valores se distinguem dos chamados graus de certeza da teoria de probabilidade clássica? Tente entender melhor isso por meio do artigo:  
[http://www.inf.ufpr.br/alex/ARTIGOS\\_IA/Nutter\\_1987.pdf](http://www.inf.ufpr.br/alex/ARTIGOS_IA/Nutter_1987.pdf)
- O que se pode dizer sobre a interface do usuário?

### Obsevações finais:

- Use apenas os compiladores POPLOG-Prolog ou SWI-Prolog;
- Use apenas predicados criados por você (não use bibliotecas do Prolog);
- Para a leitura de dados, use o predicado `read(X)` pois ele é padrão no Prolog;
- Não troque o nome do predicado “*diagnostico*” pois isso dificultará a correção do trabalho.

### Veja mais um exemplo com o monitoramento acionado:

```
?- consult(grso9).
?- assertz(monitoramento).
?- diagnostico(chamado).

>> Descobrimo sobre chamado.
>> Tentando a regra [[idade jovem] [especie humana] [sexo masculino] -> 1.0 [chamado menino]].
>> Descobrimo sobre idade.
```

```
Me informe sobre idade ? jovem.
Qual eh o seu grau de exatidao (um numero entre -1 e 1) ? 1.
```

```
>> A exatidao de idade = jovem passou a ser 1.0.
>> Descobrimo sobre especie.
>> Tentando a regra [[pernas 2] -> 1.0 [especie humana]].
>> Descobrimo sobre pernas.
```

```
Me informe sobre pernas ? 2.
Qual eh o seu grau de exatidao (um numero entre -1 e 1) ? 1.
```

```
>> A exatidao de pernas = 2 passou a ser 1.0.
>> A exatidao de especie = humana passou a ser 1.0.
>> Tentando a regra [[pernas 4] -> -1.0 [especie humana]].
>> Descobrimo sobre pernas.
>> Abandonando esta regra para especie.
```

```
>> Tentando a regra [[pernas 4] -> 0.5 [especie cao]].
>> Descobrindo sobre pernas.
>> Abandonando esta regra para especie.
>> Tentando a regra [[altura baixa] -> 0.5 [especie cao]].
>> Descobrindo sobre altura.
```

Me informe sobre altura ? baixa.

Qual eh o seu grau de exatidao (um numero entre -1 e 1) ? -0.2.

```
>> A exatidao de altura = baixa passou a ser -0.2.
>> Abandonando esta regra para especie.
>> Tentando a regra [[altura alta] -> 0.5 [especie humana]].
>> Descobrindo sobre altura.
>> Abandonando esta regra para especie.
>> Descobrindo sobre sexo.
```

Me informe sobre sexo ? masculino.

Qual eh o seu grau de exatidao (um numero entre -1 e 1) ? -0.1.

```
>> A exatidao de sexo = masculino passou a ser -0.1.
>> Abandonando esta regra para chamado.
>> Tentando a regra [[idade jovem] [especie humana] [sexo feminino] -> 1.0 [chamado menina]].
>> Descobrindo sobre idade.
>> Descobrindo sobre especie.
>> Descobrindo sobre sexo.
>> Abandonando esta regra para chamado.
>> Tentando a regra [[idade adulto] [especie humana] [sexo masculino] -> 1.0 [chamado homem]].
>> Descobrindo sobre idade.
>> Abandonando esta regra para chamado.
>> Tentando a regra [[idade adulto] [especie humana] [sexo feminino] -> 1.0 [chamado mulher]].
>> Descobrindo sobre idade.
>> Abandonando esta regra para chamado.
>> Tentando a regra [[idade jovem] [especie cao] -> 1.0 [chamado filhote]].
>> Descobrindo sobre idade.
>> Descobrindo sobre especie.
>> Abandonando esta regra para chamado.
>> Tentando a regra [[idade adulto] [especie cao] [sexo feminino] -> 1.0 [chamado cadela]].
>> Descobrindo sobre idade.
>> Abandonando esta regra para chamado.
>> Tentando a regra [[idade adulto] [especie cao] -> 0.5 [chamado cao]].
>> Descobrindo sobre idade.
>> Abandonando esta regra para chamado.
```

As hipoteses sao as seguintes: []

?- base\_de\_fatos(BF).

BF = [[sexo, [masculino, -0.1]], [altura, [baixa -0.2]], [especie, [humana 1.0]],  
[pernas, [2 1.0]], [idade, [jovem, 1.0]]] ?

yes