Segundo Trabalho de I.A. (Sistemas Especialistas) (Prof. Alexandre Direne - 2011/1)

Atenção:

Este trabalho é obrigatório e deverá ser entregue, impreterivelmente, até meio dia (12:00 horas) do dia 30 junho 2011 (quinta-feira). A solução é individual e deverá ser arquivada no diretório "~alexd/IA/" onde o nome do arquivo terá como prefixo o seu nome-de-usuário no sistema do laboratório e, como extensão, ".p" para indicar que seu conteúdo possui um programa feito com um dos pacotes do Pop-11. Assim, por exemplo, se o seu nome de usuário no sistema fosse "grs08" então o nome o arquivo seria "grs08.p" (dentro do diretório "~alexd/IA/"). Não se esqueça de proteger completamente o arquivo criado, de maneira a permitir a leitura do mesmo apenas por você! 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 a base de regras de um Sistema Especialista de classificação de erros (ver mais detalhes no parágrafo seguinte) que opera sobre assertivas representadas formalmente não só por meio de características (ou "sintomas") desses erros mas também dos aspectos de incerteza (*uncertainty*) e/ou inexatidão (*fuzziness*) associados às características em si e ao uso das mesmas para a classificação dos erros (que é o objetivo maior deste trabalho).

O domínio específico de problemas tratado pelo Sistema Especialista é o de classificação de erros de generalização que aprendizes humanos cometem no campo da Matemática denominado indução analítica sobre expressões algébricas e aritméticas. É importante destacar desde já que um erro de indução analítica pode ser mascarado por erros de naturezas algébrica e/ou aritmética. Isso causa enormes dificuldades de representação e de operação para uma máquina classificar, precisamente, as fronteiras dessas três naturezas de erros matemáticos (N.B. o que também é difícil até mesmo para professores humanos diagnosticarem em seus alunos). Antes de continuar com o entendimento deste enunciado, tente fazer os Exerícios 2 e 3 do seguinte software educacional que aborda o tópico de Progressões Geométricas em Fractais (fortemente relacionado com o tema de indução analítica) no endereço Web (use Firefox com plugin Java - evite IcedTea):

`http://condigital.c3sl.ufpr.br/fractal`

Para efeito de simplificação da classificação, apenas os erros de indução analítica serão considerados neste trabalho. Eles serão vistos de acordo com 3 (três) tipos:

1. subgeneralização;
2. supergeneralização;
3. não se aplica (é um erro de outra natureza).

Discuta com o professor o que esses três tipos realmente significam para muitas áreas do conhecimento humano pois eles também são conceitos imprecisos em suas fronteiras. No entanto, para a área de indução analítica, se você fez o Exerício 2 do software educacional citado acima, não será difícil entender o que se quer abordar aqui por meio de exemplos de instâncias desses três tipos de erros. Para a expressão correta $\frac{l}{2^n}$, alguns exemplos de erros da classe de subgeneralização são:

- $\frac{l}{2^{n+1}}$
- $\frac{l}{n^2}$
- $\frac{l}{2 \times n}$
- $\frac{l}{n}$
- $\frac{l}{2}$

Adicionalmente, para a expressão correta $\frac{l}{8}$, alguns exemplos de erros da classe de supergeneralização são:

- $\frac{l}{8 \times n}$
- $\frac{l}{n^2}$
- $3 \times \frac{l}{n}$

Cabe ainda dizer que, para a expressão correta $\frac{l}{2^n}$, alguns exemplos de erros que não se aplicam a nenhuma das duas classes citadas anteriormente são:

- $\frac{l}{16 \times 2^n}$
- $2^4 \times \frac{l}{2^n}$
- $16 \times \frac{l}{2^n}$

Para o desenvolvimento da base de regras deste trabalho, deve ser utilizado o arcabouço (*Shell*) denominado *EMYCIN* para a construção de Sistemas Especialistas do ambiente Pop-11/Poplog. Ele é um interpretador de Regras de Produção para resolver instâncias de problemas em domínios típicos de diagnóstico (*e.g.*, da área médica). Vale notar que, em um arcabouço (*Shell*) de Sistemas Especialistas (*Expert System Shell*), o estilo de representação e as estratégias de controle são sempre muito semelhantes às das *Shells* explicadas nos livros e tutoriais. Para entender detalhes da linguagem e do interpretador *EMYCIN* assim como conhecer um pouco mais sobre essas *Shells* de Sistemas Especialistas, execute:

```
source ~alexd/spop
pop11 %x im
: teach experts
```

Durante um cálculo inferencial, cada pergunta gerada pelo *EMYCIN* com a sua base de regras irá requerer do usuário tanto o valor de algum atributo como um número sintetizando o *grau de certeza* (ou de exatidão) que o usuário tem sobre o referido valor (o significado desse número é semelhante ao do número fornecido na composição da consequente de uma regra de inferência no padrão do pacote EMYCIN). Em qualquer ponto, o usuário pode perguntar "*why*" ao invés de fornecer um valor de atributo. Veja um exemplo de execução para a expressão correta $\left(\frac{l}{3^4}\right)^2 \times \frac{\sqrt{3}}{4}$ e a expressão errada $\frac{l^2}{11664} \times \sqrt{3}$:

```
: emycin("classe_de_erro" , system);
what is the quantidade_de_literais_da_expressao_incorreta_menos_da_correta ? nula
** [how sure are you ( a number between -1 and 1 ) ?]
? 1
what is the quantidade_de_operadores_exponenciacao_da_expressao_incorreta_menos_da_correta ? nula
** [how sure are you ( a number between -1 and 1 ) ?]
? 1
what is the quantidade_de_operadores_de_multiplicacao_e_divisao_da_expressao_incorreta_menos_da_correta ? negativa
** [how sure are you ( a number between -1 and 1 ) ?]
? 1
what is the quantidade_de_operadores_de_soma_e_subtracao_da_expressao_incorreta_menos_da_correta ? nula
** [how sure are you ( a number between -1 and 1 ) ?]
? 1
what is the quantidade_de_constantes_da_expressao_incorreta_menos_da_correta ? negativa
** [how sure are you ( a number between -1 and 1 ) ?]
? 0.8

** [the hypotheses are]
** [[subgeneralizacao 0.56] [nao_se_aplica 0.4]]
```

Tal diálogo resultou da tentativa de inferir um *valor* para o *atributo* `classe_de_erro` de uma "*instância*" de um dos três tipos de erros de indução analítica citados acima.

Em um programa *EMYCIN*, a base de regras de classificação se encontra todo em uma única variável denominada `system`, mas que pode ser mudado para representar o dianóstico em outros domínios quaisquer de conhecimento (*e.g.*, para classificar instâncias de doenças infecciosas). Na verdade, as representações do exemplo acima sobre classes de erros de indução analítica são apenas fragmentos que incluem alguns detalhes combinando atributos com valores qualitativos e quantitativos, tais como:

- Diferença entre a quantidade de literais da expressão incorreta e da correta;

- Diferença entre a quantidade de operadores de exponenciação (literais no expoente) da expressão incorreta e da correta;
- Diferença entre a quantidade de operadores de multiplicação e/ou divisão da expressão incorreta e da correta;
- Diferença entre a quantidade de operadores de soma e/ou subtração da expressão incorreta e da correta.
- Diferença entre a quantidade de constantes de soma e/ou subtração da expressão incorreta e da correta.

Observação: não será preciso implementar as funções que calculam o valor de cada um dos fatores acima. Ao invés disso, os valores de tais fatores serão tipicamente qualitativos e poderão ser calculados mentalmente por quem operar o seu Sistema Especialista. Ou seja, eles serão fornecidos como dados na entrada padrão, caso algum deles seja requisitado pelo sistema especialista durante sua execução. Para ver muitos exemplos reais dos três tipos de erros de indução analítica, visite a seguite página Web (use apenas o Firefox por causa de uma biblioteca de visualização de expressões matemáticas):

http://www.youse.com.br/mestrado_gustavo/DissertationSystem/src/ClusterResult.php?filename=../data/subDataClassified2Cluster.arff
http://www.youse.com.br/mestrado_gustavo/DissertationSystem/src/ClusterResult.php?filename=../data/supDataClassified2Cluster.arff

Para compor seu arquivo ".p" com a solução deste trabalho, tome como modelo o seguinte formato geral:

```
;;; ---------------------------------------------------------------------
  /*
    1) Dar e-mail de contato frequente (serÃ¡ usado na correÃ§Ã£o!);
    2) Adicionar um exemplo de dialogo com o programa;
    3) Lista cada caracteristica acompanhada de seu dominio de valores.
  */

uses emycin;

;;; Limiar de certeza ou exatidao para o encadeamento de inferencias.
;;; Tente altera-la para ver o efeito sobre o raciocinio artificial.
0.25 -> cutoff;

;;; Variavel que controla se serao emitidas mensagens de monitoramento.
;;; Altere-a para true se voce quiser ver o monitoramento.
false -> chatty;

;;; Base de regras propriamente dita.
[
  [ [...   ...]    ...    [...   ...]    =>   ...  [...   ...] ]
  [ [...   ...]    ...    [...   ...]    =>   ...  [...   ...] ]
  [ [...   ...]    ...    [...   ...]    =>   ...  [...   ...] ]
  ...
] -> system;
;;; ---------------------------------------------------------------------
```

A varável global denominada `cutoff` guarda um número entre -1.0 e 1.0. Com ele, o processo de diagnóstico *abandonará* a avaliação propagativa de qualquer regra cujo menor grau de certeza das subcláusulas de sua antecedente tenha atingido um valor menor ou igual que o da varável `cutoff`. Utilize 0.25 como um bom valor para este trabalho.

Os valores da escala de certeza ou 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 o *valor-verdade* da varável global denominada `chatty`. Veja um outro exemplo com a varável `chatty` valendo `true`, no qual a expressão correta é $\frac{l}{3^4}$ e a expressão errada é $\frac{l}{4 * n}$:

```
: true -> chatty;
: emycin("classe_de_erro" , system);
** [finding out about classe_de_erro]
** [trying rule [[quantidade_de_literais_da_expressao_incorreta_menos_da_correta negativa] => 1 [classe_de_erro subgeneralizacao]]]
** [finding out about quantidade_de_literais_da_expressao_incorreta_menos_da_correta]
what is the quantidade_de_literais_da_expressao_incorreta_menos_da_correta ? positiva
** [how sure are you ( a number between -1 and 1 ) ?]
? 1
** [the certainty of quantidade_de_literais_da_expressao_incorreta_menos_da_correta being positiva is now 1]
** [abandoning this rule for classe_de_erro]
** [trying rule [[quantidade_de_literais_da_expressao_incorreta_menos_da_correta positiva] => 1 [classe_de_erro supergeneralizacao]]]
** [finding out about quantidade_de_literais_da_expressao_incorreta_menos_da_correta]
** [the certainty of classe_de_erro being supergeneralizacao is now 1]
** [trying rule [[quantidade_de_literais_da_expressao_incorreta_menos_da_correta nula] [quantidade_de_operadores_exponenciacao_da_expressao_incorreta_menos_da_corre
    ta nula] [quantidade_de_operadores_de_multiplicacao_e_divisao_da_expressao_incorreta_menos_da_correta negativa] [quantidade_de_operadores_de_soma_e_subtracao_da
    _expressao_incorreta_menos_da_correta nula] [quantidade_de_constantes_da_expressao_incorreta_menos_da_correta negativa] => 0.7 [classe_de_erro
    subgeneralizacao]]]
** [finding out about quantidade_de_literais_da_expressao_incorreta_menos_da_correta]
** [abandoning this rule for classe_de_erro]
** [trying rule [[quantidade_de_literais_da_expressao_incorreta_menos_da_correta nula] [quantidade_de_operadores_exponenciacao_da_expressao_incorreta_menos_da_corre
    ta nula] => 0.4 [classe_de_erro nao_se_aplica]]]
** [finding out about quantidade_de_literais_da_expressao_incorreta_menos_da_correta]
** [abandoning this rule for classe_de_erro]

** [the hypotheses are]
** [[supergeneralizacao 1]]
```

A *Base de Fatos* (veja a teoria de Sistemas de Produção) é utilizada para armazenar informações sobre os atributos, seus valores e seus graus de certeza ou exatidão. 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 certeza de 0.3 sobre a altura da instância ser `baixa` e uma certeza de -0.4 sobre a altura da instância ser `alta`. O código da Base de Fatos se encontra na variável denominada `database`. Veja um exemplo de Base de Fatos depois que terminar uma inferência sobre outro exemplo:

```
: database ==>
[[altura, [baixa, 0.2]], [especie, [humana, 1.0]], [pernas, [2, 1.0]]]
```

Além dos livros já recomendados nas transparências da disciplina, para ler mais sobre incerteza e inexatidão, veja:

```
http://eprints.cs.vt.edu/archive/00000042/01/TR-86-36.pdf
```

Para ler mais ainda sobre Sistemas Especialistas e pacotes de regras de produção, veja:

```
source ~alexd/spop
pop11 %x im
: teach prodsys;
: teach newsys;
```

Se você se interessar em conhecer o código Pop-11 do próprio interpretador $EMYCIN$ (*i.e.*, da *Shell*), basta fazer:

```
: showlib emycin
```

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?
- Como o conhecimento está representado?
- Quais são os principais componentes do programa criado?
- Como as inferências estão organizadas?
- Quais são os papéis exercidos pelas "*probabilidades*" (graus de certeza ou exatidão) na repesentação do conhecimento e no processo de inferência?
- O que se pode dizer sobre a interface do usuário?

Obsevações finais:

- Use apenas o compilador POPLOG;
- Não troque o nome do atributo "`classe_de_erro`" pois isso dificultará a correção do trabalho.