Inteligência Artificial
IA

Prof. João Luís Garcia Rosa

III. PROVA AUTOMÁTICA DE TEOREMAS

2004

Representação do conhecimento

- Suponha o seguinte corpo de conhecimento (exemplo 1):
  1. Marco era um homem.
  2. Marco era um pompeiano.
  3. Todos os pompeianos eram romanos.
  4. César era um soberano.
  5. Todos os romanos ou eram leais a César ou o odiavam.
  6. Todos são leais a alguém.
  7. As pessoas somente tentam assassinar soberanos aos quais elas não são leais.
  8. Marco tentou assassinar César.

Representação do conhecimento

- Representando este conhecimento através de fórmulas da lógica de primeira ordem:
  1. $homem(marco)$
  2. $pompeiano(marco)$
  3. $\forall X\ (pompeiano(X) \rightarrow romano(X))$
  4. $soberano(césar)$
  5. $\forall X\ (romano(X) \rightarrow (leal(X,césar) \vee odiar(X,césar)))$
  6. $\forall X \exists Y\ leal(X,Y)$
  7. $\forall X \forall Y\ ((pessoa(X) \wedge tentarassassinar(X,Y) \wedge soberano(Y)) \rightarrow \neg leal(X,Y))$
  8. $tentarassassinar(marco,césar)$

Representação do conhecimento

- Suponha que se deseje usar este conhecimento para responder à questão
  - “Marco era leal a César?”
- Parece que usando 7 e 8, dá para concluir que Marco não era leal a César (ignorando a distinção entre passado e presente).
- Há a necessidade de inclusão de conhecimento de senso comum:

  9. Todos os homens são pessoas.

  $\forall X\ (homem(X) \rightarrow pessoa(X))$

Representação do conhecimento

- Deste exemplo simples, pode-se perceber três pontos importantes, na conversão de sentenças do português em fórmulas da lógica:
  - Muitas sentenças do português são ambíguas (por exemplo, 5, 6 e 7 acima). A escolha da interpretação correta pode ser difícil.
  - Existe freqüentemente uma escolha de como representar o conhecimento. Representações simples são desejáveis mas elas podem impedir certos tipos de raciocínio.
  - Mesmo em situações muito simples, um conjunto de sentenças não parece conter toda a informação necessária para raciocinar sobre o tópico em questão. Para ser capaz de usar um conjunto de fórmulas efetivamente, é muitas vezes necessário ter acesso a um outro conjunto de fórmulas que representam fatos considerados óbvios demais para mencionar (senso comum).

Representação do conhecimento

- Um outro problema surge em situações onde não se conhece de antemão quais comandos deduzir. No exemplo apresentado, o objeto era responder a questão “Marco era leal a César?” Como um programa poderia decidir se ele deveria tentar provar

  leal(marco,césar)

- ou

  ¬leal(marco,césar)

Representação do conhecimento

- Funções e predicados computáveis
  - Predicados comuns
    tentarassassinar(marco,césar)
  - Predicados computáveis “maior-que” e “menor-que”:

ma(1,0)	me(0,1)
ma(2,1)	me(1,2)
ma(3,2)	me(2,3)

  - Funções computáveis:
    ma(mais (2,3),1)

Representação do conhecimento

- Exemplo 2:

1. Marco era um homem.
2. Marco era um pompeiano.
3. Marco nasceu em 40 D.C.
4. Todos os homens são mortais.
5. Todos os pompeianos morreram quando o vulcão entrou em erupção em 79 D.C.
6. Nenhum mortal vive mais de 150 anos.
7. Agora é 2004.
8. Vivo significa não morto.
9. Se alguém morre, então ele está morto para sempre.

Representação do conhecimento

1. *homem(marco)*
2. *pompeiano(marco)*
3. *nascer(marco,*40*)*
4. $\forall X\ (homem(X) \rightarrow mortal(X))$
5. $erupção(vulcão,79) \wedge \forall X(pompeiano(X) \rightarrow morreu(X,79))$
6. $\forall X \forall T1 \forall T2((mortal(X) \wedge nasceu(X,T1) \wedge ma(T2-T1,150)) \rightarrow morto(X,T2))$
7. *agora* = 2004
8. $\forall X \forall T((vivo(X,T) \rightarrow \neg morto(X,T)) \wedge (\neg morto(X,T) \rightarrow vivo(X,T)))$
9. $\forall X \forall T1 \forall T2\ ((morreu(X,T1) \wedge ma(T2,T1)) \rightarrow morto(X,T2))$

Representação do conhecimento

- A partir da base de conhecimento pode-se tentar responder à questão "Marco está vivo agora?" provando:

  $\neg vivo(marco,agora)$

- Vai-se estudar formas automáticas de se realizar estas provas: **Prova Automática de Teoremas**.

Unificação

- Um par (*X*,*T*) é uma *substituição simples* (lê-se "*X* substituído por *T*") se e somente se *X* é uma variável e *T* é um termo.
- Um conjunto finito β de substituições simples é uma *substituição* se e somente se duas substituições simples em β não coincidem no primeiro elemento.
- Uma substituição β é uma *substituição básica* se e somente se, para todo (*X*,*T*) em β, *T* é um termo sem ocorrências de variáveis.
- β é a *substituição vazia "ε"* se e somente se β for o conjunto vazio.
- Uma substituição β é uma *renomeação de variáveis* ou, simplesmente, uma *renomeação*, se e somente se cada par (*X*,*T*) em β for tal que *T* é uma variável e não existirem dois pares (*X*,*U*) e (*Y*,*V*) em β tais que $X \neq Y$ e $U = V$.

Unificação

- Uma *expressão* é qualquer seqüência de símbolos de um alfabeto de primeira ordem, e uma *expressão simples* é qualquer literal ou termo sobre o alfabeto.
- Seja C uma cláusula e β uma substituição. A *instanciação* de C por β, denotada por Cβ, é a cláusula obtida instanciando-se C por β e eliminando-se as ocorrências repetidas do mesmo literal, exceto a ocorrência mais à esquerda.

- A *composição* de substituições é a função, denotada por "°", que mapeia pares de substituições em uma substituição e é definida da seguinte forma:
  - Para todo par de substituições
    $\beta = \{X1/T1, ..., Xn / Tn, Y1/S1, ..., Yk / Sk\}$
    $\theta = \{Y1/R1, ..., Yk / Rk, Z1 / Q1, ..., Zm / Qm\}$
  - onde $X1, ..., Xn$, $Y1, ..., Yk$, $Z1, ..., Zm$ são variáveis distintas, a composição de β com θ será a substituição:
    $\beta \circ \theta = \{X1 /(T1)\theta, ..., Xn /(Tn)\theta, Y1 /(S1)\theta, ..., Yk /(Sk)\theta, Z1 / Q1, ..., Zm / Qm\}$

- Exemplo: Seja β = {X/f(Y), Y/Z} e θ = {X/a, Y/b, Z/Y}. Então a composição das substituições β com θ é a substituição
  β ° θ = {X/f(b), Y/Y, Z/Y}
- que é obtida, aplicando-se θ aos termos das substituições simples em β, formando o conjunto {X/f(b), Y/Y} e acrescentando a única substituição simples de θ, "Z/Y", cujo primeiro elemento não coincide com o primeiro elemento de uma substituição simples em β.
- *Propriedades*:
  - $\theta \circ \varepsilon = \varepsilon \circ \theta = \theta$
  - $(E\theta)\beta = E(\theta \circ \beta)$, qualquer que seja a expressão E
  - Se $E\theta = E\beta$ então $\theta = \beta$, qualquer que seja a expressão E
  - $(\theta \circ \beta) \circ \varphi = \theta \circ (\beta \circ \varphi)$

Unificação

- Seja C = l1 l2 ... uma cláusula com conjunto de literais **L** = {l1, l2} e β uma substituição.
  - β é um *unificador* para **L** se e somente se (l1)β = (l2)β.
  - β é um *unificador mais geral* (ou, abreviadamente, um *u.m.g.*) de **L** se e somente se β é um unificador de **L** e, para todo unificador θ de **L**, existe uma substituição φ tal que θ = β ° φ.
  - conjunto **L** é *unificável* se e somente se existe um unificador para **L**.

Algoritmo da Unificação

- Um conjunto de termos **D** é o *conjunto de discórdia* de um conjunto de literais **L** = {l1 ,...,l*n* } se e somente se
  - **D** = ∅, se *n* = 1;
  - **D** = {T1 ,...,T*n* }, se *n* > 1 e todas as expressões em **L** são idênticas até o *i*-ésimo símbolo, exclusive, e T*j* é o termo ocorrendo em **L** que começa no *i*-ésimo símbolo, para *j* = 1,...,*n*.
- Uma substituição simples *X*/*T satisfaz o teste de ocorrência* se e somente se *X* não ocorre em *T*.
- Um conjunto de discórdia **D** *satisfaz o teste de ocorrência* se e somente se existem uma variável *X* e um termo *T* em **D** tais que *X* não ocorre em *T*.

Algoritmo da Unificação

Exemplo: Sejam os seguintes conjuntos de literais e seus conjuntos de discórdia:

conjunto de literais	*conjunto de discórdia*
{p(a, X, f(g(Y))), p(Z, h(Z,W), f(W))}	{a,Z}
{p(a, X, f(g(Y))), p(a, h(a,W), f(W))}	{X, h(a,W)}
{p(a, h(a,W),f(g(Y))), p(a, h(a,W), f(W))}	{g(Y), W}

Em todos os exemplos acima, o conjunto de discórdia satisfaz o teste de ocorrência.

Algoritmo da Unificação

- *entrada*: - um conjunto **L** de literais
- *saída*: - um u.m.g. β de **L**, se **L** for unificável
  - 'NÃO', se **L** não for unificável

início
β := ε;
W := **L**;
D := conjunto de discórdia de W;
enquanto (número de literais de W) > 1 e **D** satisfaz o teste de ocorrência *faça*
    início
    selecione uma variável *X* e um termo *T* em **D** tais que *X* não ocorre em *T*;
    β := β ° {*X*/*T*};
    W := W{*X*/*T*};
    D := conjunto de discórdia de W;
    fim;
se (número de literais de W) = 1
    então retorne β
    senão retorne 'NÃO'
fim

- *O que é resolução*
  - O sistema formal da resolução trabalha exclusivamente com **cláusulas** e contém apenas uma regra de inferência, chamada de **regra da resolução**, que gera uma nova cláusula a partir de duas outras. Dado um conjunto **S** de cláusulas e uma cláusula C, uma dedução de C a partir de **S** neste sistema formal consiste de uma seqüência de cláusulas terminando em C e gerada aplicando-se repetidamente a regra da resolução. Uma refutação a partir de **S** é uma dedução da cláusula vazia a partir de **S**. A regra da resolução é definida de tal forma que **S** é insatisfatível se e somente se existe uma refutação a partir de **S**.

- Na definição da regra da resolução, tratar-se-á uma cláusula não-vazia “l1 ...l*n*” como o conjunto finito {l1 ,...,l*n* } e a cláusula vazia “ ” como o conjunto vazio. Assim, utilizar-se-á as operações usuais de teoria dos conjuntos para definir novas cláusulas a partir de outras. Por exemplo, se “l m n” e “n p” são cláusulas, a expressão “(l m n) ∪ (n p)” denota a cláusula “l m n p” (a ordem dos literais no resultado é irrelevante em face da semântica das cláusulas).

- Uma cláusula A é uma *instância* de B se e somente se existir uma substituição β = {*X*1/*T*1, ..., *Xn* /*T*n } de variáveis por termos tal que A é obtida substituindo-se simultaneamente *Xi* por *Ti* em B, para *i* = 1,...,*n*. Usar-se-á Bβ para denotar o resultado da substituição.
  – Exemplo: Seja a cláusula B = p(*X*) q(*X*,*Y*). Seja uma substituição β = {*X*/a,*Y*/f(b)}. A instanciação de B por β, denotada por Bβ, é a cláusula instância A = p(a) q(a,f(b)).

- A regra da resolução combina:
  – uma adaptação para cláusulas da regra *Modus Ponens* (regra R1)
  – um processo de "unificação" de literais de duas cláusulas (regra R2)
  – um processo de "unificação" de literais de uma mesma cláusula (regra da resolução RE).

- **Regra R1:** se A' possui um literal l e A" possui um literal ¬l, derive A = (A' - l) ∪ (A" - ¬l).
  - Exemplo: Seja o seguinte conjunto de cláusulas:
    1. p(*X*) ¬q(*Y*)
    2. q(*Y*) r(*Z*)
  - dá para obter, usando a regra R1, a cláusula
    3. p(*X*) r(*Z*)

- Seja agora o seguinte conjunto de cláusulas:
  1. p(*X*) ¬q(*Y*)
  2. q(*W*) r(*Z*)
- não é possível mais obter a cláusula 3, pois as variáveis são diferentes. A regra R2 vai resolver este problema.

- **Regra R2:** se A' possui um literal l' e A" possui um literal ¬l" e existe uma substituição β tal que l'β = l"β, derive A = (A'β - l'β) ∪ (A"β - ¬l"β)

Resolução

– <u>Exemplo</u>: Retomando o conjunto anterior

1. p(*X*) ¬q(*Y*)
2. q(*W*) r(*Z*)

– existe uma substituição β = { *W*/*Y* }, que aplicada às duas cláusulas, resulta no seguinte

1. p(*X*) ¬q(*Y*)
2. q(*Y*) r(*Z*)

– que obviamente produz a cláusula abaixo

3. p(*X*) r(*Z*)

Resolução

- O processo de tornar idênticos os literais em uma cláusula C através de uma substituição de variáveis por termos é chamado de **unificação** e a substituição é chamada de um **unificador** de C. Um **unificador mais geral** é aquele que, intuitivamente, especifica as substituições mais simples possíveis. O processo de unificação deverá então utilizar sempre um unificador mais geral para não bloquear outras unificações.

- Por exemplo, $\beta = \{X/f(a), Y/a\}$ e $\theta = \{X/f(Y)\}$ unificam C = p(X) p(f(Y)) pois $C\beta$ = p(f(a)) e $C\theta$ = p(f(Y)). Porém θ é um unificador mais geral do que β. A substituição θ é então preferível a β pois, por exemplo, o literal "p(f(b))" é unificável com o literal em $C\theta$, mas não é unificável com o literal em $C\beta$.

- Diz-se que uma cláusula B é um **fator** de uma cláusula A se e somente se existe um conjunto **L** de literais de A e existe um unificador mais geral β para **L** tal que B = $A\beta$. Note que uma cláusula A é um fator dela mesma. O processo de obter fatores de cláusulas é chamado de **fatoração**.

- **Regra RE:** se B' e B" são fatores de cláusulas A' e A" tais que B' possui um literal l' e B" um literal ¬l" e existe um unificador mais geral β para l' e l", derive A = (B'β - l'β) $\cup$ (B"β - ¬l"β)
  - Neste caso, diz-se que a cláusula A é um *resolvente* de A' e A", que são as *cláusulas pais*.

Resolução

– <u>Exemplo</u>: Seja o seguinte conjunto de cláusulas:

1. p(*X*) q(*Z*)
2. ¬r(*Y*) p(*T*) ¬r(*W*)
3. r(*V*) ¬q(*U*)

4. p(*T*) ¬r(*W*) ¬q(*U*) R2: 2,3 $\beta = \{V/Y\}$
5. p(*X*) p(*T*) ¬r(*W*) R2: 1,4 $\beta = \{Z/U\}$

2'. ¬r(*Y*) p(*T*) fator de 2, com $\beta = \{W/Y\}$

4. p(*T*) ¬q(*U*) RE: 2',3 $\beta = \{Y/V\}$
5. p(*X*) RE: 1, 4 $\beta = \{U/Z\}$ e fator de p(*X*) p(*T*)

Resolução

- Em resumo, o sistema formal de resolução trabalha apenas com cláusulas, que são objetos com uma sintaxe bem simples, e possui apenas uma regra de inferência, a regra da resolução. Um procedimento de **refutação baseado em resolução** é então um procedimento que, dado um conjunto qualquer de cláusulas **S**, procura sistematicamente derivar a cláusula vazia utilizando apenas a regra da resolução e tendo como ponto de partida as cláusulas em **S**. A construção de tais procedimentos requer, porém, métodos especiais para tentar contornar a explosão combinatorial gerada pela liberdade de escolha de cláusulas, fatores e literais.