AGENTES LÓGICOS

Sistemas Inteligentes

Agosto/2006

Agentes

- O conhecimento dos agentes de resolução de problemas é muito específico e inflexível
  - Exemplo: agente jogador de xadrez
- Os agentes baseados em conhecimento
  - Podem combinar e recombinar informações para atender a uma infinidade de propósitos
    - Exemplo: combinar conhecimento geral com percepções correntes para deduzir aspectos ocultos
  - O conhecimento e o raciocínio
    - Crucial ao lidar com ambientes parcialmente observáveis

Agentes baseados em conhecimento

- Ex: Um médico faz o diagnóstico de um paciente antes do tratamento
  - Doença não é diretamente observável
  - Conhecimento
    - Uma parte em forma de regras aprendidas com livros e professores
    - Uma parte em forma de padrões de associação

Agentes baseados em conhecimento

- Ex: Linguagem natural
  - Exige dedução do estado oculto - a intenção do falante
  - “John viu a jóia pela janela e **a** desejou”
  - “John lançou a pedra contra a janela e **a** quebrou”
  - Conhecimento comum – problemas com agentes de resolução de problemas
  - Problemas de contingência são ineretemente exponenciais

Agentes baseados em conhecimento

- Tem flexibilidade
- São capazes de aceitar novas tarefas sob a forma de metas descritas de modo explícito
- Pode alcançar competência rapidamente
  - Ao serem informados ou
  - Adquirirem novos conhecimentos sobre o ambiente
- Podem se adaptar a mudanças no ambiente
  - Atualizando o conhecimento relevante

Representação de conhecimento

- A lógica é o princípio da maioria da tecnologias de representação de conhecimento
- Exemplo simples de representação p/ agentes baseados em conhecimento
- Apresenta algumas limitações severas:
  - Não consegue representar muito bem nem incerteza, nem tempo
- Lógica proposicional e Lógica de primeira ordem - LPO

Agentes baseados em conhecimento

- Componente central – Base de conhecimento (BC)
  - É um conjunto de sentenças
    - Expressas uma linguagem de representação de conhecimento
    - Representam alguma asserção sobre o mundo
- Para adicionar sentenças à BC – TELL (informe)
- Para consultar o que se conhece – ASK (pergunte)
- TELL e ASK podem envolver inferência: derivação de novas sentenças a partir de sentença antigas

Agente baseado em conhecimento genérico

Função AGENTE-BC(*percepção*) **retorna** uma *ação*

variáveis estáticas: *BC*, uma base de conhecimento

t, um contador, inicialmente igual a zero contando o tempo

TELL(*BC*, CRIAR-SENTENÇA-DE-PERCEPÇÃO(*percepção*,*t*))

ação <= ASK(*BC*, CRIAR-CONSULTA-DE-AÇÃO(*t*)) raciocínio

TELL(*BC*, CRIAR-SENTENÇA-DE-AÇÃO(*ação*, *t*))

t <= *t* + 1

retornar *ação*

Agentes baseados em conhecimento

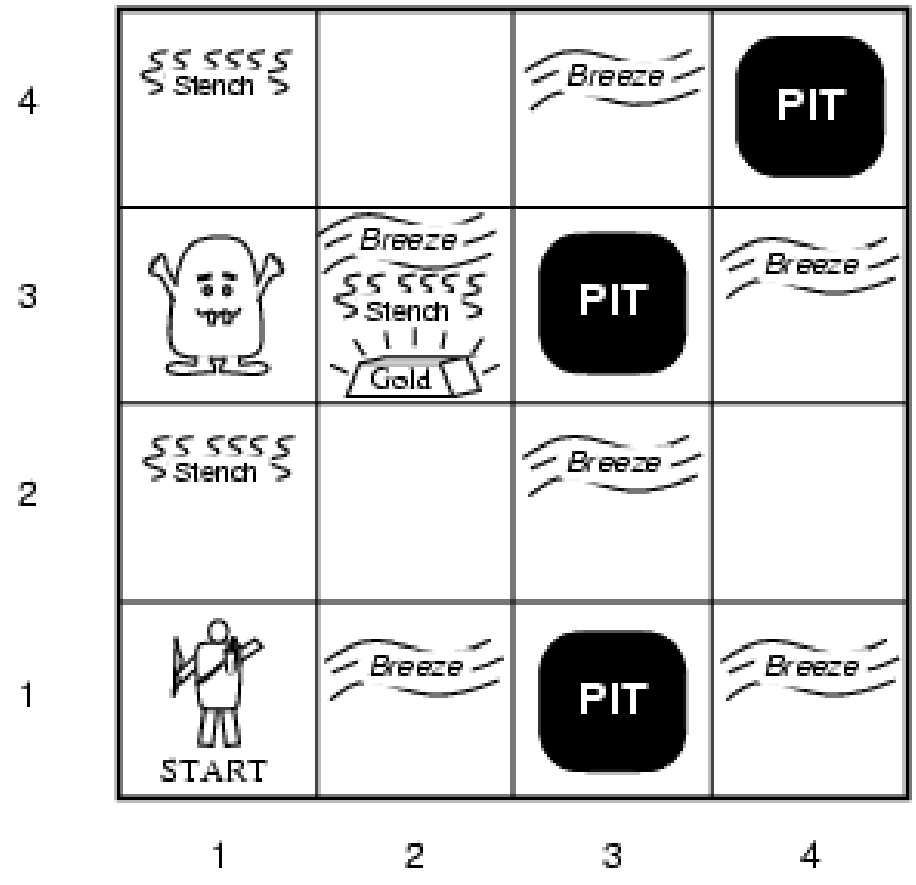
- Semelhantes aos agentes com estado interno
- Mas não é um programa arbitrário para calcular ações
- Nível de conhecimento
  - Precisamos especificar apenas o que o agente sabe e quais são suas metas
- Nível de implementação
  - Não estamos preocupados como o conhecimento é implementado: listas, mapas, strings, neurônios

Construindo um agente baseado em conhecimento

- Informamos (com TELL) o que ele precisa conhecer
- Também podemos fornecer mecanismos que lhe permitam aprender por si mesmo
  - Mecanismos criam conhecimento geral sobre o ambiente a partir de uma série de percepções
  - Este conhecimento pode ser incorporado à BC do agente e usado na tomada de decisões

O mundo do Wumpus

- Wumpus e fedor nos adjacentes
- Poços e brisa nos adjacentes
- Ouro e brilho no local
- Atirar mata o Wumpus se o agentes estiver virado para ele
- O agente só tem uma flecha
- Agente morre se entrar em uma sala com um poço ou o Wumpus vivo



O mundo do Wumpus

- Medida de desempenho
  - Pegar o ouro = +1000
  - Cair em poço ou ser devorado = -1000
  - Executar ação = -1
  - Usar a flecha = -10

- Ambiente
  - Agente começa em [1,1], voltado para a direita
  - Posições do ouro e do Wumpus são aleatórias
  - Cada sala pode ter um poço com probb. 0,2

O mundo do Wumpus

- Atuadores
  - Agente pode se mover para frente, ou a esquerda e a direita 90º
  - Mover-se para frente não tem efeito se houver uma parede
  - Ação agarrar – pega um objeto que está no mesmo quadrado que o agente
  - Ação atirar – atira a flecha em linha reta diante do agente
    - A flecha irá parar somente quando atingir o Wumpus ou a parede

O mundo do Wumpus

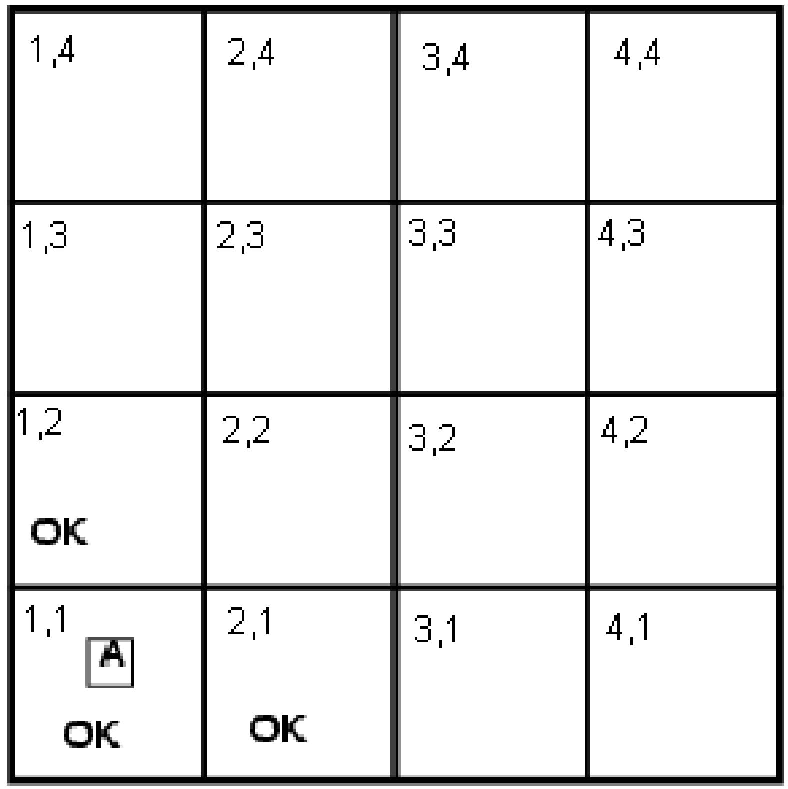
- Sensores: fornecem um único bit de informação
  - Fedor – quadrado com o Wumpus e adjs. a ele
  - Brisa – quadrados adjs. a um poço
  - Brilho – quadrado do ouro
  - Impacto – quando caminhar para uma parede
  - Grito – do Wumpus quando morre
  - Exemplo: [Fedor, Brisa, Nada , Nada , Nada]

- Principal dificuldade – ignorância inicial da configuração do ambiente – exige exploração e raciocínio

Explorando o mundo do Wumpus

- Inicialmente a BC contém somente as regras do ambiente
- Em particular o agente sabe que está em [1,1] e este quadrado é seguro
- A 1ª percepção: [Nada, Nada, Nada, Nada, Nada]
- Então o agente pode concluir que os quadrados adjacentes são seguros

Explorando o mundo do Wumpus

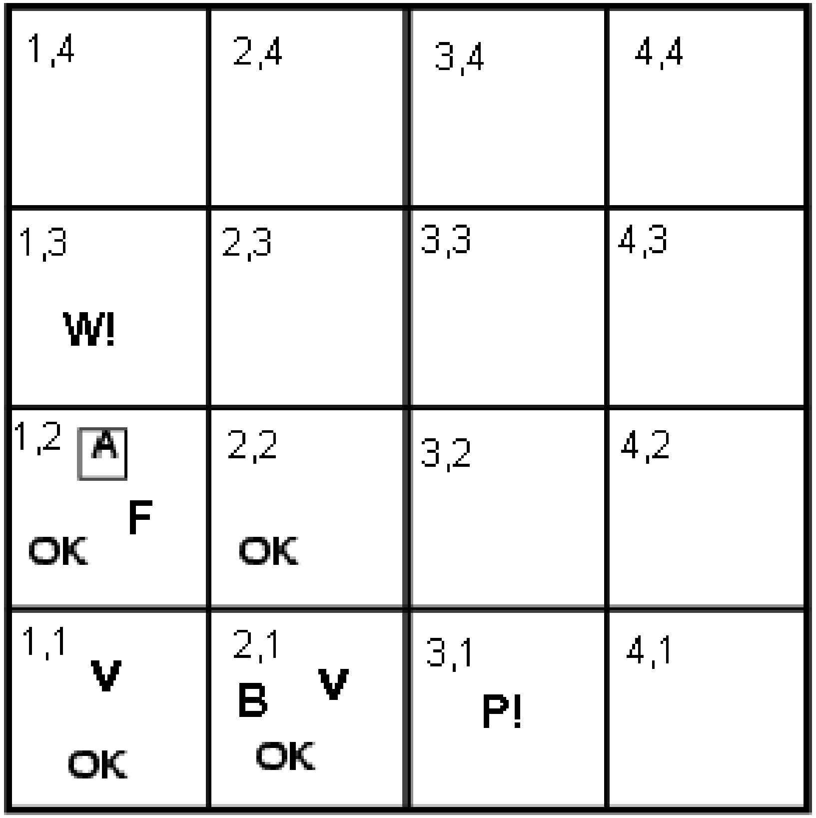


Explorando o mundo do Wumpus

1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 B $\boxed{A}$ OK	3,1 P?	4,1

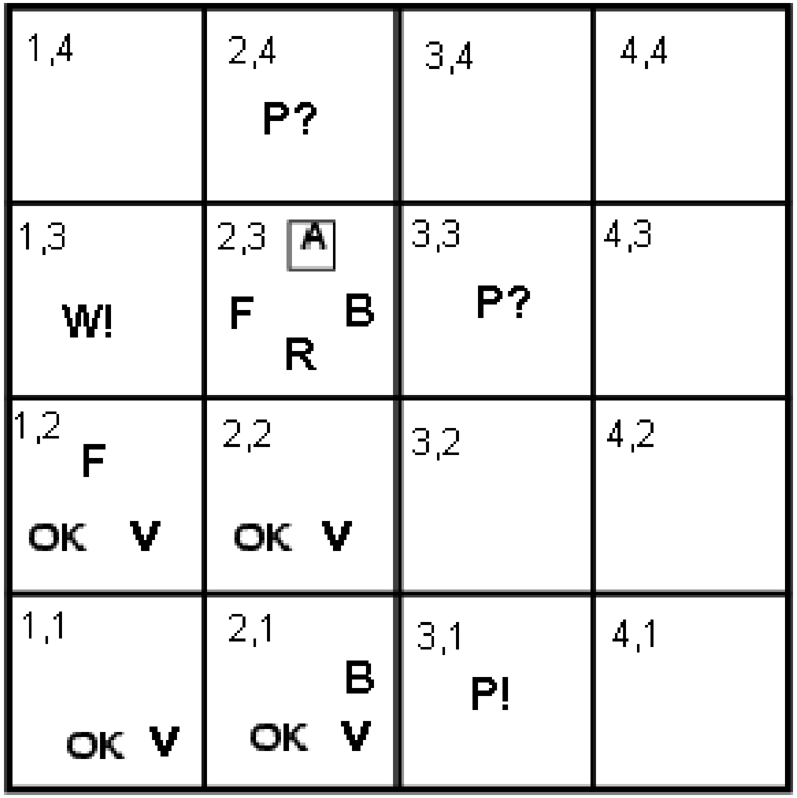
Explorando o mundo do Wumpus

- Se o poço estivesse em [2,2], haveria brisa em [1,2]
- Fedor em [1,2]
- O wumpus pode estar em [2,2] ou [1,3]
- Mas se estivesse em [2,2], haveria fedor em [2,1]
- Wumpus em [1,3]



Explorando o mundo do Wumpus

- O único quadrado seguro é [2,2]
- Não existe B em [2,2]
- Então [2,3] e [3,2] são seguros
- O agente vai para [2,3]
- F em [2,3], confirmando que o Wumpus está em [1,3]
- B em [2,3] poços em [2,4] ou [3,3]
- R em [2,2], ouro em [2,2]
- Pega o ouro e encerra.



Lógica

- Como representar as informações do wumpus e tirar conclusões sobre elas?

- As sentenças da BC são expressas de acordo com a sintaxe da ling. de representação de conhecimento

- Sintaxe especifica as sentenças que são bem-formadas
  - X + Y= 4 (bem-formada)
  - X2Y + = (não é bem-formada)

Lógica

- Semântica da lógica = significado das sentenças

- Em lógica a semântica define a verdade de cada sentença em relação a cada mundo possível (modelo)
  - X + Y = 4
    - Verdadeira em um mundo em que X=2 e Y=2
    - Falsa em um mundo em que X=1 e Y=1

- Em lógicas-padrão ou uma sentença é verdadeira ou falsa
  - Não existe um meio termo

Lógica Proposicional ou Booleana

- É muito simples
- Sintaxe
  - Sentenças atômicas – elementos sintáticos indivisíveis – consistem em um único símbolo proposicional
  - Cada símbolo representa uma sentença que pode ser verdadeira ou falsa
  - Nomes arbitrários, como mnemônicos
    - $W_{1,3}$ para representar que o Wumpus está em [1,3]
  - Símbolos com significados fixos: **verdadeiro** e **falso**

Lógica Proposicional

- Sintaxe
  - Sentenças complexas são construídas a partir de sentenças simples + conectivos lógicos
  - Conectivos lógicos
    - $\neg$ (não) – **negação** de uma sentença. Ex: $\neg W_{1,3}$
    - $\wedge$(e) – **conjunção** de duas sentenças. Ex: $W_{1,3} \wedge P_{3,1}$
    - $\vee$(ou) – **disjunção** de duas sentenças. Ex: $P_{3,1} \vee P_{2,2}$

Lógica Proposicional

- Conectivos lógicos
  - => **implicação**. Ex**:** $(W_{1,3} \wedge P_{3,1}) \Rightarrow \neg W_{2,2}$
    - Onde $(W_{1,3} \wedge P_{3,1})$ é chamado premissa ou **antecedente**
    - E $\neg W_{2,2}$ é chamado conclusão ou **conseqüente**
  - <=> se e somente se – **bicondicional**
    - Ex: $W_{1,3} \Leftrightarrow \neg W_{2,2}$
  - Ordem de precedência (decrescente) dos conectivos
    - ¬, ∨ ,∧<=> ,<= ,
    - $\neg P \vee Q \wedge R \Rightarrow S \quad \rightarrow \quad ((\neg P) \vee (Q \wedge R)) \Rightarrow S$

Lógica Proposicional

- Semântica
  - Define as regras p/ especificar a verdade de uma sentença em relação a um modelo específico
  - Um modelo fixa um valor verdade para um símbolo proposicional
    - Se a BC é composta dos símbolos: $P_{1,2}$, $P_{2,2}$ e $P_{3,1}$
    - Então m_1 = {$P_{1,2}$=falsa, $P_{2,2}$=falsa, e $P_{3,1}$=verdadeira} é um modelo possível
    - Três símbolos proposicionais = 2^3 modelos possíveis

Lógica Proposicional

- Semântica
  - Deve especificar como calcular o valor verdade de qualquer sentença, dado um modelo
  - Sentenças complexas são formadas de sentenças atômicas + conectivos
  - Sentenças atômicas
    - *Verdadeiro* é verdadeiro e *falso* é falso em todo modelo
    - O valor verdade dos outros símbolos proposicionais são especificados no modelo.
      - Ex: $P_{1,2}$ é falsa no modelo do exemplo

Lógica Proposicional

- Semântica
  - Conectivos lógicos
    - As regras para cada um podem ser resumidas em uma tabela-verdade

P	**Q**	**¬P**	**P ∧ Q**	**P ∨ Q**	**P => Q**	**P <=> Q**
F	F	V	F	F	V	V
F	V	V	F	V	V	F
V	F	F	F	V	F	F
V	V	F	V	V	V	V

    - $\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,2}) \rightarrow$ Verd ∧ (False ∨ Verd) = Verd

Lógica Proposicional

- BC = conjunto de sentenças
- BC lógica é uma conjunção destas sentenças
- Se a BC estiver vazia e fizermos
  - TELL(BC,S_1), TELL(BC,S_2) , ..., TELL(BC,S_n)
  - Teremos BC = $S_1 \land S_2 \land \ldots \land S_n$

Lógica Proposicional

- Mundo do Wumpus
  - As regras são mais bem escritas usando-se <=>
    - Um quadrado tem brisa se e somente se um quadrado vizinho tem poço
    - $B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$
  - A implicação simples $B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})$ é verdadeira, mas incompleta
    - Ela não elimina modelo em que $B_{1,1}$ é falsa e $P_{1,2}$ é verdadeira

Uma BC para o mundo do Wumpus

- Vamos falar somente sobre os poços
- Vocabulário de símbolos proposicionais
  - Seja $P_{i,j}$ verdadeira se existe um poço em [i,j]
  - Seja $B_{i,j}$ verdadeira se existe brisa em [i,j]
- A BC é formada pelas sentenças
  - Não há nenhum poço em [1,1]
    - S_1: $\neg P_{1,1}$

Uma BC para o mundo do Wumpus

- Um quadrado tem brisa se e somente se existe um poço em um quadrado vizinho
  - S_2: $B_{1,1} <=> (P_{1,2} \vee P_{2,1})$
  - S_3: $B_{2,1} <=> (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
  - Isto deve ser declarado para cada quadrado

- As sentenças acima são verdadeiras para todos os mundos de wumpus

- As percepções de brisa nos dois primeiros quadrados do exemplo:
  - S_4: $\neg B_{1,1}$
  - S_5: $B_{2,1}$

Uma BC para o mundo do Wumpus

- A BC consiste em R_1 a R_5

- Ou pode ser considerada uma conjunção
  - $R_1 \land R_2 \land R_3 \land R_4 \land R_5$

- Que afirma que todas as sentenças individuais são verdadeiras

Lógica Proposicional - Fraquezas

- A lógica proposicional é suficiente para ilustrar os conceitos básicos da lógica
- Mas é muito fraca para representar ambientes complexos de forma concisa
  - Ex: para descrever o mundo do Wumpus somos forçados a escrever uma regra separada p/ brisas e poços p/ cada quadrado

Lógica de Primeira Ordem - LPO

- A LPO é suficientemente expressiva para representar de forma satisfatória nosso conhecimento comum

- É o alicerce de muitas outras linguagens de representação

- Empresta da linguagem natural a idéia de:
  - Objetos – substantivos e sentenças nominais
  - Relações – verbos e sentenças verbais
    - Funções – relações com um só valor p/ uma dada entrada
    - Propriedades – relações unárias

LPO

- Exemplos:
  - Objetos: Pessoas, casas, números, teorias, séculos...
  - Relações: irmão de, maior que, interior a, parte de...
  - Funções: Pai de, melhor amigo, início de...
  - Propriedades: vermelho, redondo, falso, primo...
- LPO pode expressar fatos sobre alguns ou todos os objetos no universo
- Permite representar leis ou regras gerais como:
  - “Quadrados vizinhos a poços tem brisa”

LPO

- Lógica proposicional: existem fatos que são válidos ou não são válidos no mundo

- LPO: o mundo consiste em objetos com certas relações entre eles que são ou não são válidas

Linguagens Formais

Linguagem	Comp. Ontológico (O que existe no mundo)	Comp. Epistemológico (a crença do agente sobre os fatos)
Lógica Proposicional	Fatos	Verdadeiro/falso/desconhecido
LPO	Fatos, objetos, relações	Verdadeiro/falso/desconhecido
Lógica Temporal	Fatos, objetos, relações, tempos	Verdadeiro/falso/desconhecido
Teoria da probb	Fatos	Grau de crença entre 0 e 1
Lógica difusa	Fatos com graus de verdade entre 0 e 1	Valor de intervalo conhecido

LPO

- Modelos em lógica proposicional:
  - Conjuntos de valores-verdades para os símbolos proposicionais.
- Modelos em LPO:
  - Contém objetos.
  - O domínio de um modelo é o conjunto de objetos que ele contém.
  - Objetos também chamados de elementos do domínio.

LPO – Exemplo de modelo

- Objetos:
  - Ricardo Coração de Leão (rei da Inglaterra 1189-1199)
  - Rei João (rei da Inglaterra 1199-1215)
    - Irmão mais novo de Ricardo
    - Perverso
  - Perna esquerda de Ricardo
  - Perna esquerda de João
  - Uma Coroa

LPO – Exemplo de modelo

- Relações:
  - Uma relação é um conjunto de tuplas de objetos inter-relacionados.
  - Relações binárias:
    - Relação de fraternidade do modelo
      - <Ricardo Coração de Leão, Rei Jõao>
      - <Rei Jõao, Ricardo Coração de Leão>
    - Relação “na cabeça”
      - <coroa, Rei João>

LPO – Exemplo de modelo

- Relações unárias ou propriedades:
  - “pessoa” -> verdadeira para Ricardo e João
  - “rei” -> verdadeira para João
  - “coroa” -> verdadeira para a coroa

- Relações – funções unárias:
  - “perna esquerda”
    - <Ricardo Coração de Leão> -> perna esquerda de Ricardo
    - <Rei João> -> perna esquerda de João

LPO – Sintaxe

- Símbolos que representam objetos, relações e funções:
  - Símbolos de constantes – representam objetos.
    - Ex: *Ricardo* e *João*
  - Símbolos de predicados – representam relações.
    - Ex: *Irmão*, *NaCabeça*, *Pessoa*, *Rei* e *Coroa*
  - Símbolos de funções – representam funções.
    - Ex: *PernaEsquerda*
  - *Onde a escolha dos nomes cabe ao usuário do modelo.*

LPO – Semântica

- Relaciona sentenças a modelos para determinar a verdade.
- Interpretação possível para o exemplo = interpretação pretendida
  - *Ricardo* se refere a Ricardo Coração de Leão
  - *João* se refere ao perverso rei João
  - *Irmão* se refere à relação de fraternidade
  - *NaCabeça* se refere à relação "na cabeça" que é valida entre a coroa e o rei João
  - *Pessoa*, *Rei* e *Coroa* se referem a objetos que são pessoa, reis e coroas
  - *PernaEsquerda* se refere á função "perna esquerda"

LPO – Sintaxe

Sentença -> SentençaAtômica
| (Sentença conectivo Sentença)
| Quantificador Variável, ... Sentença
| ¬Sentença
SentençaAtômica -> predicado(Termo, ...) | Termo=Termo
Termo -> Função(Termo, ...)
| Constante
| Variável
Conectivo -> $\vee$ | $\wedge$ | <=> | =>
Quantificador -> $\forall$ | $\exists$
Constante -> A | X_1 | ...
Variável -> a | x | s | ...
Predicado -> Antes | TemCor | Chovendo | ...
Função -> Mãe | Pai | PernaEsquerda | ...

LPO - Termos

- É uma expressão lógica que se refere a um objeto.
  - Símbolos constantes são termos
  - Símbolos de função também.
    - *PernaEsquerda*(*João*)

- Termos complexos:
  - Um símbolo de função seguido por uma lista de argumentos entre parênteses.
  - É apenas uma espécie complicada de nome.
    - Não é uma chamada de subrotina que retorna um valor.
    - Não existe uma função “PernaEsquerda” que recebe uma pessoa como entrada e retorne uma perna.

LPO – Sentença Atômica

- Enunciam fatos.

- É formada por um símbolo de predicado, seguido por um lista de termos entre parênteses:
  - *Irmão*(*Ricardo*, *João*)
  - Ricardo coração de Leão é irmão do rei João

- Podem ter termos complexos como argumentos:
  - *Casado*(*Pai*(*Ricardo*), *Mãe*(*João*))
  - Ricardo Coração de Leão é casado com a mãe do rei João

LPO – Sentenças Complexas

- São construídas utilizando conectivos lógicos.
- A semântica é idêntica à da lógica proposicional.
- Exemplo de sentenças verdadeiras:
  - $\neg Irmão(PernaEsquerda(Ricardo), João)$
  - $Irmão(Ricardo, João) \wedge Irmão(João, Ricardo)$
  - $Rei(Ricardo) \vee Rei(João)$
  - $\neg Rei(Ricardo) => Rei(João)$

LPO - Quantificadores

- Nos permite expressar propriedade de coleções inteiras de objetos, em vez de enumerar todos pelo nome.
- Dois tipos:
  - Universal ($\forall$)
  - Existencial ($\exists$)

Quantificador Universal (∀)

- Lógica proposicional – dificuldade em expressar regras gerais:
  - “Quadrados vizinhos ao wumpus são fedorentos”
  - “Todos os reis são pessoas”

- Em LPO este tipo de regra é comum:
  - $\forall x\ Rei(x) \Rightarrow Pessoa(x)$
  - Para todo *x*, se *x* é um rei, então *x* é uma pessoa.
  - *x* é chamado de variável – e também é um termo.
  - $\forall x$ ***P***, onde P é qualquer expressão lógica
    - Afirma que ***P*** é verdadeira para todo objeto *x.*

Quantificador Universal (∀)

- Considere o modelo do exemplo. Podemos estender a interpretação "instanciando" *x* para cada objeto:
  - *x* -> Ricardo Coração de Leão
  - *x* -> rei João
  - *x* -> perna esquerda de Ricardo
  - *x* -> perna esquerda de João
  - *x* -> A coroa

- $\forall x\ Rei(x) \Rightarrow Pessoa(x)$ é verdadeira sob a interpretação original se ela for verdadeira em cada uma das sentenças estendidas.

Quantificador Universal ($\forall$)

- Assim a sentença $\forall x\ Rei(x) \Rightarrow Pessoa(x)$ é equivalente a afirmar que:
  - Ricardo Coração de Leão é um rei => Ricardo Coração de Leão é uma pessoa
  - O rei João é um rei => O rei João é uma pessoa
  - A perna esquerda de Ricardo é um rei => perna esquerda de Ricardo é uma pessoa
  - A perna esquerda de João é um rei => perna esquerda de João é uma pessoa
  - A coroa é um rei => A coroa é uma pessoa

Quantificador Universal (∀)

- Observando a TV para a implicação temos:

P	Q	P => Q
F	F	V
F	V	V
V	F	F
V	V	V

- Perfeito para escrever regras gerais com ∀.

- Um erro muito comum é escrever:
  - $\forall x\ Rei(x) \land Pessoa(x)$

Quantificador Existencial (∃)

- Permite declarar algo sobre algum objeto do domínio sem nomeá-lo.

- Exemplo:
  - $\exists x\ Coroa(x) \wedge NaCabeça(x, João)$
  - O rei João tem uma coroa em sua cabeça.

- $\exists x\ P$ afirma que *P* é verdadeira para pelo menos um objeto x do domínio.

- $\exists x\ P$ é verdadeira sob uma dada interpretação se:
  - *P* é verdadeira em pelo menos uma interpretação estendida que atribui *x* a algum objeto do domínio.

Quantificador Existencial (∃)

- $\exists x\ Coroa(x) \land NaCabeça(x, João)$ afirma que pelo menos uma das afirmações deve ser verdadeira:
  - Ricardo Coração de Leão é uma coroa ∧ Ricardo Coração de Leão está na cabeça de João
  - O rei João é uma coroa ∧ O rei João está na cabeça de João
  - A perna esquerda de Ricardo é uma coroa ∧ a perna esquerda de Ricardo está na cabeça de João
  - A perna esquerda de João é uma coroa ∧ a perna esquerda de João está na cabeça de João
  - A coroa é uma coroa ∧ a coroa está na cabeça de João

Quantificador Existencial (∃)

- Como vimos, ∀ deve ser utilizado com =>, e
- No caso do ∃, deve ser utilizado com o ∧.
- Pois o uso do ∃ com o => em geral conduz a uma declaração muito fraca:
  - $\exists x\ Coroa(x) \Rightarrow NaCabeça(x, João)$
  - Assim a sentença é verdadeira em todo modelo que contém um objeto para qual a premissa é falsa.
  - Não contempla nosso objetivo de dizer que existe pelo menos um objeto coroa que está na cabeça de João.

Inferência em LPO

- Componente fundamental – Unificação.
- Serve para descobrir substituições que façam expressões lógicas diferentes parecerem idênticas.
- O algoritmo de unificação recebe duas sentenças e retorna um unificador para elas, se existir algum.
- Como exemplo suponha que temos uma consulta à BC *Conhece*(*João*, *x*) -> quem conhece João?

Inferência em LPO - Unificação

- Suponha que a BC dispõe das sentenças:
  - *Conhece*(*João*, *Jane*)
  - *Conhece*(*y*, *Bill*)
  - *Conhece*(*y*, *Mãe*(*y*))
  - *Conhece*(*x*, *Elizabeth*)

- Resultados da Unificação:
  - UNIFICAR(*Conhece*(*João*, *x*), *Conhece*(*João*, *Jane*)) = {*x*/*Jane*}
  - UNIFICAR(*Conhece*(*João*, *x*), *Conhece*(*y*, *Bill*)) = {*x*/*Bill*, *y*/*João*}
  - UNIFICAR(*Conhece*(*João*, *x*), *Conhece*(*y*, *Mãe*(*y*))) = {*y*/*João*, *x*/*Mãe*(*João*)}
  - UNIFICAR(*Conhece*(*João*, *x*), *Conhece*(*x*, *Elizatbeth*)) = falha

Inferência em LPO - Unificação

- A unificação:
  - UNIFICAR(*Conhece*(*João*, *x*), *Conhece*(*x*, *Elizatbeth*)) falha somente porque *x* não pode assumir os valores *João* e *Elizabeth* ao mesmo tempo.

- Pois a sentença *Conhece*(*x*, *Elizabeth*) significa que "Todo mundo conhece Elizabeth" – inclusive o João.

- O problema só surge pq as duas sentenças utilizam o mesmo nome de variável, *x*.

- O problema pode ser evitado renomeando as variáveis de uma das duas sentenças – padronização separada.

Inferência em LPO - Unificação

- A unificação:
  - UNIFICAR(*Conhece*(*João*,*x*), *Conhece*(*y*, *z*))

- Poderia retornar:
  - {*y*/ *João*, *x*/*z*} -> *Conhece*(*João*, *z*) com resultado
  - {*y*/ *João*, *x*/*João*, *z*/*João*} -> *Conhece*(*João*, *João*)

- O primeiro unificador é mais geral que o segundo, pq impõe menos restrições sobre os valores das variáveis.

- Para todo par de expressões que deve ser unificado, existe um único unificador mais geral ou UMG.

Exemplo

- A lei diz que é crime um americano vender armas para nações hostis.
- O país nono, um inimigo da América, tem alguns mísseis
- Todos estes mísseis foram vendidos para ele pelo Coronel West, que é Americano

- Prove que o Coronel West é um criminoso.

Exemplo BC:

- ... é crime para um americano vender armas para nações hostis:
  - $Americano(x) \land Arma(y) \land Vende(x,y,z) \land Hostil(z) \Rightarrow Criminoso(x)$
- Nono… tem alguns mísseis:
  - $\exists x\ Proprietário(Nono,x) \land Míssil(x)$
  - $Proprietário(Nono,M1) \land Míssil(M1)$
- ... todos estes mísseis foram vendidos para ele pelo Coronel West:
  - $Míssil(x) \land Proprietário(Nono,x) \Rightarrow Vende(West,x,Nono)$

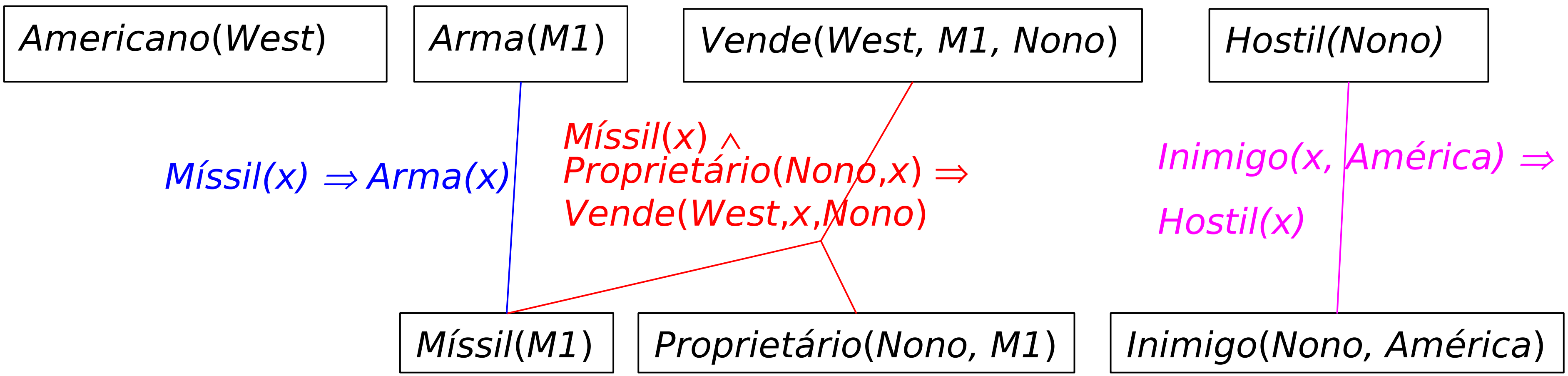
Exemplo BC:

- Mísseis são armas:
  - $Míssil(x) \Rightarrow Arma(x)$
- Um inimigo da América é considerado hostil:
  - $Inimigo(x, América) \Rightarrow Hostil(x)$
- West, que é Americano …
  - $Americano(West)$
- O país Nono, é um inimigo da América
  - $Inimigo(Nono, América)$

Prova da Base para a Conclusão

Míssil(*M1*)	*Proprietário*(*Nono*, *M1*)	*Inimigo*(*Nono*, *América*)

Prova da Base para a Conclusão



Prova da Base para a Conclusão