Lógica de Primeira Ordem

Capítulo 8

Lógica Proposicional

- Linguagem declarativa ➔ sua semântica se baseia em uma relação-verdade entre sentenças e mundos possíveis
  - Capacidade de expressão para lidar com informações parciais, usando disjunções e negação
  - Linguagem composicional: o significado de suas sentenças é uma função do significado de suas partes. Exemplo: a verdade de $\alpha \wedge \beta$ depende da verdade de α e de β

Lógica Proposicional (LP)

- Problemas:
  - Falta de capacidade de expressão para descrever de forma concisa um ambiente com muitos objetos. Exemplo:

$$B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$$
$$B_{1,2} \Leftrightarrow (P_{1,3} \lor P_{2,2})$$
$$B_{1,3} \Leftrightarrow (P_{1,2} \lor P_{1,4} \lor P_{2,3})$$
...

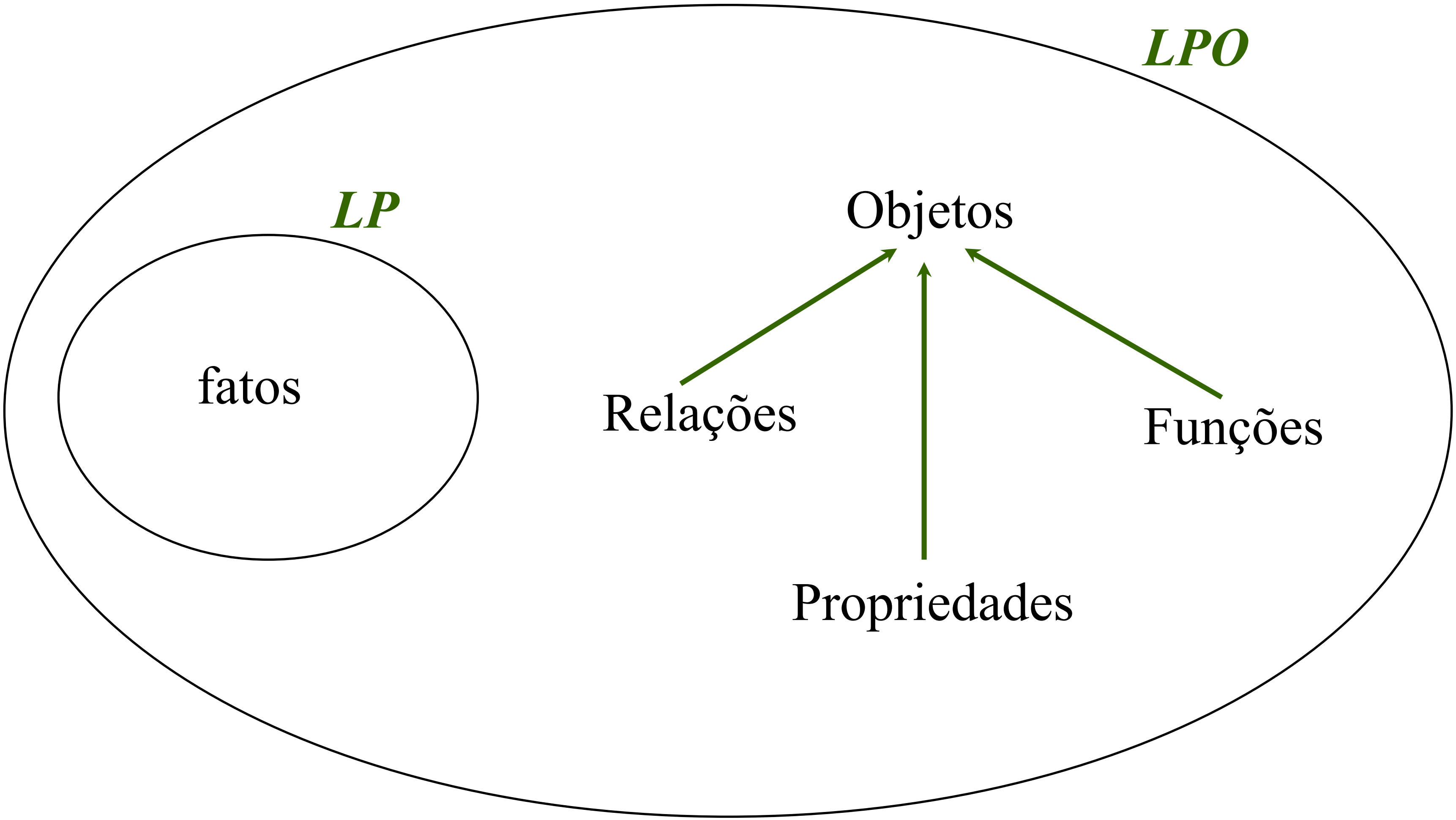
  - Linguagem natural: "*existe brisa nos quadrados adjacentes a poços*"

Lógica de Primeira Ordem (LPO)

Faz a suposição que o mundo é constituido de ***objetos*** com certas ***propriedades*** ou ***relações*** entre eles.

Objetos podem ser definidos em ***função*** de outros objetos

LP versus LPO (FOL)



LP versus LPO

Compromissos ontológicos:

- LP: existem fatos que são verdadeiros ou falsos no mundo
- LPO: o mundo é constituido de objetos com certas relações entre eles que são verdadeiras ou falsas
- Outro exemplo de compromisso ontológico:
  - Lógica temporal: fatos são verdadeiros falsos em instantes específicos do tempo e esses instantes (pontos ou intervalos) estão ordenados

Exemplos de objetos, relações e funções

- **Objetos**: pessoas, casas, números, Lula, cores, anos, guerras, ...
- **Relações unárias** (ou **propriedades**): vermelho, redondo, falso, primo, ...
- **Relações n-árias**: irmão de, maior que, tem cor, pertence a, fica entre, ...
- **Funções**: pai de, melhor amigo, terceiro

Objetos, relações e funções

"Quadrados vizinhos ao Wumpus são fedorentos"

- **Objetos**: *quadrados*, *Wumpus*
- **Propriedade**: *fedorentos*
- **Relação**: *vizinhos*

Modelos em LP e LPO

- LP: valores verdade atribuidos para os símbolos proposicionais que ...

  ... tornam sentenças da KB verdadeiras
- LPO: modelos contêm objetos. O domínio de um modelo é o conjunto de objetos que ele contém. Os objetos dos modelos podem estar relacionados de várias maneiras e é para essas relações que são atribuidos valores verdade que ...

  ... tornam sentenças da KB verdadeiras

LPO

- Em LPO um literal contém **termos**

  LP: $W_{1,2}$

  LPO: Wumpus(***linha***, ***coluna***)

  – Termos são expressões construídas a partir de ***símbolos constantes***, ***variáveis*** e ***símbolos de funções***.

  – Termos são referências aos objetos do modelo

- **Símbolos constantes**: *A, B,C, João, Rei, Presidente, Chefe, Professor*. Cada constante nomeia um único objeto.

- **Variáveis**: *x*, *y*, *casa*, *percepção*, *linha* … . Cada variável nomeia um conjunto de objetos.

LPO - Sintaxe

Símbolos de predicados (relações)

– Especificam uma relação entre os objetos de um modelo. Um predicado pode ser binário ou n-ário.

– Ex.: *Irmão*(Lendro, Leonardo), *Irmão*(Leonardo, Leandro). *Irmão* se refere ao conjunto de todas as duplas de objetos do modelo:

{ <Leandro, Leonardo>,
<Leonardo, Leandro>,
<ZezeDeCamargo, Luciano>,
<Luciano, ZezeDeCamargo>, …>}

LPO - Sintaxe

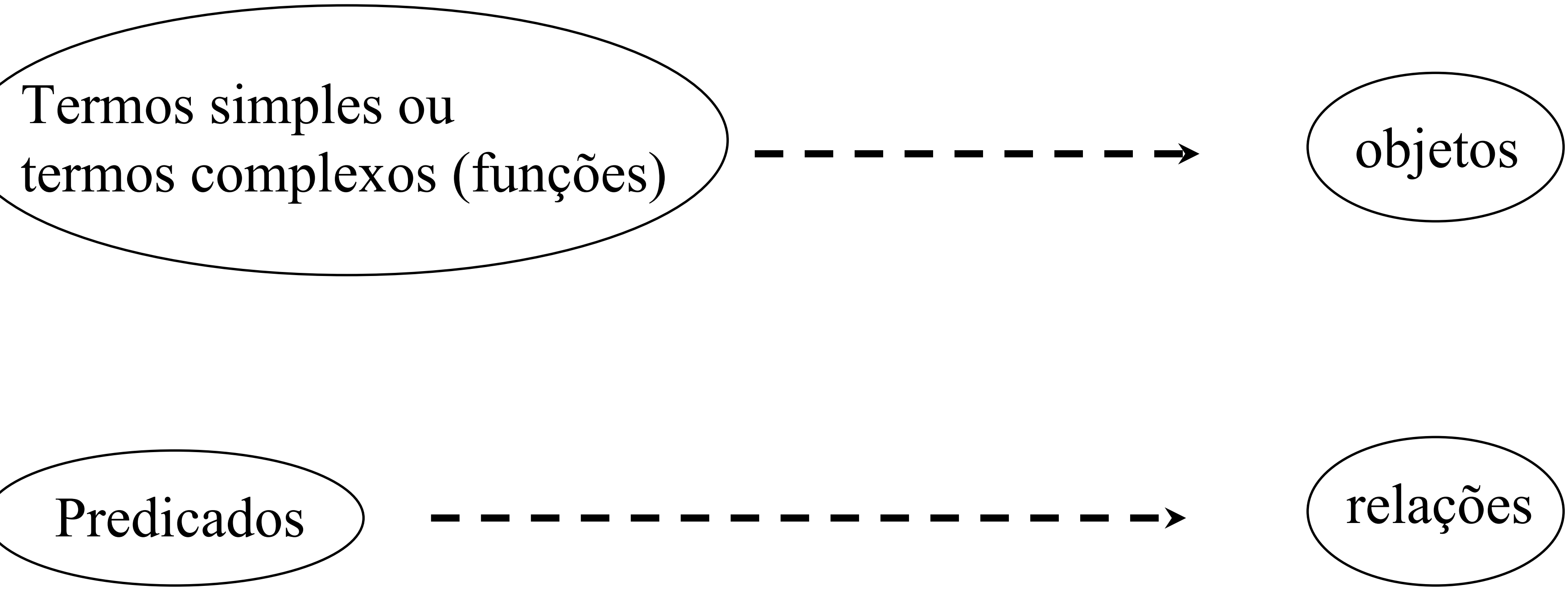
Símbolos de funções:

– Algumas relações entre objetos são *funcionais*: relacionam objetos com um outro objeto do modelo (def. de função).

– Exemplos: *PaiDe(Leandro), IrmãoDe(Leonardo), Idade(Leo), Altura(Ana)*...

– No modelo: conjunto de (n+1)-uplas, sendo o último elemento, o valor da função para os n primeiros elementos

Interpretação



LPO - Sintaxe

- **Sentenças atômicas**:

  Declaram fatos. Formadas por um símbolo predicado seguida por uma lista de termos entre parênteses

  irmão(Leandro, Leonardo)

  é-pai(João, Pedro) (≠ função: pai-de(João))

  A verdade de uma sentença atômica depende da interpretação e do modelo

- **Sentenças complexas**:

  Uso dos conectivos lógicos. A semântica das sentenças é a mesma da LP

LPO - Sintaxe

- **Quantificadores:** usados para expressar propriedades de coleções inteiras de objetos, ao invés de enumerar cada um dos objetos pelo seu nome. LPO possue dois quantificadores:
  - **Quantificador Universal ($\forall$)**
  - **Quantificador Existencial ($\exists$)**

Quantificador Universal (∀)

- Exemplo:
  - “ Todos os gatos são mamíferos ”
    - Gato(Mimi) ^
    - Gato(Felix) ^
    - Mamífero(Mimi) ^
    - Mamífero(Felix) ^…

$\forall$ x Gato(x) => Mamífero(x)

“Para todo x, se x é um gato então x é mamífero”

Quantificador Universal ($\forall$)

$\forall$ ***x*** *P*, onde *P* é qualquer sentença lógica, afirma que *P* é verdadeira para todo objeto *x*

Gato(Mimi) $\Rightarrow$ Mamífero(Mimi) ^

Gato(Felix) $\Rightarrow$ Mamífero(Felix) ^

Gato(João) $\Rightarrow$ Mamífero(João) ^

Gato(Pedro) $\Rightarrow$ Mamífero(Pedro) ^

....

… define que todas essas sentenças são verdadeiras!

Quantificador Existencial (∃)

Permite escrever uma sentença com objetos particulares sem citar o seu nome:

“ *Felix tem uma irmã que é uma gata*”

$\exists\, x\ Irmã(x,Felix) \wedge Gata(x)$

“*Existe x, x é irmã de Felix e x é uma gata*”

Quantificador Existencial (∃)

$\exists$ ***x*** *P*, onde *P* é verdade para algum objeto no universo, por exemplo:

$$P \equiv \exists\, x\ Irmã(x,Felix)\ \wedge\ Gata(x)$$

Irmã(Mimi, Felix) ^ Gato(Mimi) v

Irmã(Felix, Felix) ^ Gato(Felix) v

Irmã(João, Felix) ^ Gato(João) v

Irmã(Pedro, Felix) ^ Gato(Pedro) v

…

… define que pelo menos uma dessas sentenças é verdadeira

Quantificadores aninhados

$\forall\, x\, \forall\, y\;\; Pai(x,y) \Rightarrow Filho(y,x)$

$\forall\, x,\, y\;\; Irmão(x,y) \Rightarrow Irmão(x,y)$

- "Todo mundo ama alguém"

  $\forall\, x\, \exists\, y\; Ama(x,y)$

- "Existem pessoas que são amadas por todas"

  $\exists\, y\, \forall\, x\; Ama(x,y)$

- A ordem dos quantificadores é importante

$$\forall\, x\, (\exists\, y\; P(x,y)) \;\neq\; \exists\, x\, (\forall y\; P(x,y))$$

Conexões entre $\exists$ e $\forall$

$$\forall x \, \neg Gosta(x, Cenouras)$$

é equivalente a

$$\neg \exists x \, Gosta(x, Cenouras)$$

$$\forall x \, Gosta(x, Cenouras)$$

é equivalente a

$$\neg \exists x \, \neg Gosta(x, Cenouras)$$

Igualdade

- Outra maneira de se construir sentenças atômicas em LPO:
  - uso do símbolo de igualdade para fazer com que dois termos correspondam ao mesmo objeto

$$\text{Pai(João)} = \text{Pedro}$$

predicado

  - Um outro exemplo:

$$\exists\, x, y\ \mathit{Irmão}(x,\mathit{Ricardo}) \wedge \mathit{Irmão}(y,\mathit{Ricardo}) \wedge \neg(x = y)$$

Lógica de Primeira Ordem com Igualdade

LPO - Sintaxe

Sentença => SentençaAtômica |
(Sentença Conectivo Sentença) |
Quantificador Variável Sentença |
$\neg$ Sentença |

SentençaAtômica => Predicado(*Termo, ...)* |
Termo = Termo

Termo => Função(termo) |
Constante |
Variável

Conectivo => $\wedge$ | $\vee$ | $\rightarrow$ | $\leftrightarrow$

Quantificador => $\forall$ | $\exists$

Constante => A | X1 | João | …

Variável => a | x | s | ...

Predicados => Antes | TemCor | Chovendo | …

Função => Mão | irmão | Resultado

Asserções e consultas em LPO

TELL(KB,Rei(João))

TELL(KB, $\forall$ *x Rei(x)* $\rightarrow$ *Pessoa(x))*

ASK(KB,Rei(João)) ?

... devolve *verdadeiro*.

ASK(KB,Pessoa(João)) ?

... devolve *verdadeiro*.

Asserções e consultas em LPO

ASK(KB, ∃ x Pessoa(x)) ?

... devolve *verdadeiro*.

- Esse tipo de consulta também pode fornecer um valor para x. Chamamos de *substituição* ou *lista de unificações* a um conjunto de pares variável/termos

ASK(KB, ∃ x Pessoa(x)) ?

... devolve {*x/João*}.

Unificação

Processo de encontrar uma substituição de variáveis (λ) que faz com que duas sentenças se tornem iguais

- Unifica(p,q) = λ sendo Subst(λ,p) = Subst(λ,q)

Exemplos:

- Unifica(Conhece(João,x), Conhece(x, Maria)) = *falha*
  - x não pode ser igual a João e Maria ao mesmo tempo
- Unifica(Conhece(João,x1), Conhece(x2, Maria) = (x1/Maria, x2/João)
- Unifica(Conhece(x1,x2), Conhece(João, Maria) = (x1/João, x2/Maria)

Extensões e variações notacionais

- Lógicas de ordem maior: podem quantificar relações e não somente os objetos do domínio. Por exemplo:

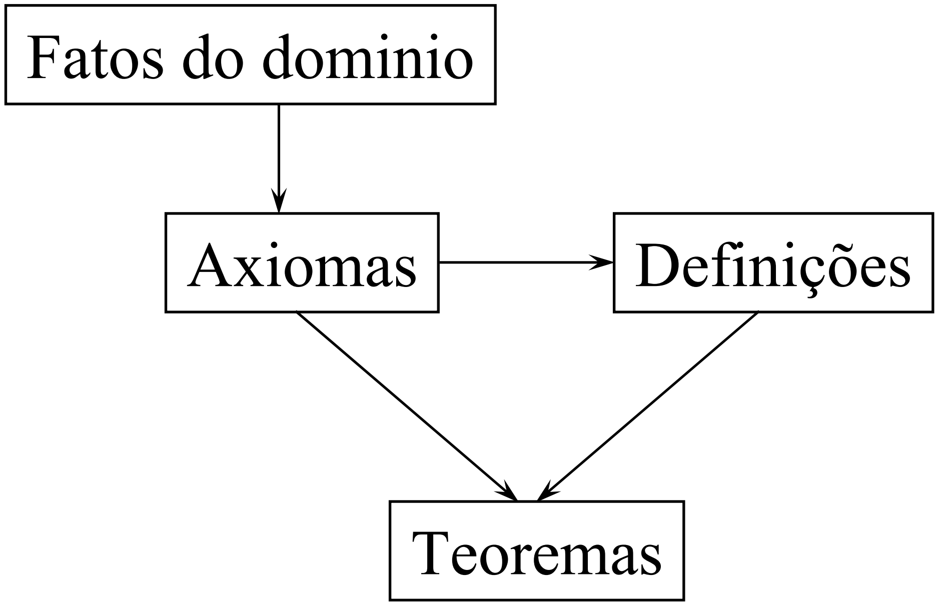
$$\forall\, x,y\ (x=y) \Leftrightarrow (\forall p\ p(x) \Leftrightarrow p(y))$$

"dois objetos são iguais sse se todas as propriedades aplicadas a eles são equivalentes", ou

$$\forall\, f,g\ (f=g) \Leftrightarrow (\forall x\ f(x) = g(x))$$

"duas funções são iguais sse elas possuirem o mesmo valor para todos os argumentos

Axiomas, Definições e Teoremas



Axiomas capturam fatos básicos sobre um domínio $\Rightarrow$ usados para definir conceitos $\Rightarrow$ usados para provar teoremas

Axiomas, definições e teoremas

- Qual é a quantidade de axiomas necessária para definir um domínio?
- Qual é a quantidade de predicados necessária para definir um domínio?
- Existem muitos axiomas? Exemplo de escolhas na modelagem de um domínio:

$$\forall x \exists y \; Irmãos(x,y) \Leftrightarrow Irmãos(y,x)$$

ou

$$\exists p \; Pai(p,João) \wedge Pai(p,Pedro)$$

(Exercício 7.6)

Axiomas, definições e teoremas

- ***Axiomas***: sentenças que não podem ser derivadas de outros axiomas
- ***Teoremas*****:** sentenças que podem ser derivadas dos axiomas
- Matemáticos se esforçam para produzir um conjunto mínimo de axiomas que sejam independentes entre si (i.e., um não pode ser derivado a partir do outro).
- IA usa teoremas para tornar o processo de busca mais eficiente

Exemplo: Domínio de parentesco

Domínio: parte do mundo do qual queremos representar o conhecimento

Domínio da realeza

Elizabeth é uma mulher

Elizabeth é a rainha

Charles é um homem

Elizabeth é mãe de Charles

Charles é pai de William

Se x é a mãe de y e y é pai de z, então x é avó de z

(Página 246. Exercício 8.11)

O domínio dos números, conjuntos e listas

- Constante: *Conjunto-vazio*
- Predicados: *Membro-de*, *Subconjunto*
- Funções: *Intersecção, União, Adiciona-elemento*
- ….. Sentenças de uma linguagem de conjuntos: pag.248

(Exercício 8.14)

Mundo de Wumpus - FOL

- FOL permite construir um agente-KB com maior capacidade de raciocínio e desempenho (maior poder de representação)
- Três arquiteturas de agentes lógicos:
  - **agente reativo** (classificação de percepção e ação)
  - **agentes baseados em modelo** (possuem representação interna do mundo)
  - **agentes baseados em meta** (especificam metas e tentam satisfazê-las)

Percepções e ações

- Sentenças deverão incluir percepção e o tempo que ocorreram. O agente deve guardar todas as suas percepções do presente e passado
- Uma sentença típica seria:

  Percebe([*Cheiro, Brisa, Brilho, Nada, Nada*],5)
- Ações como termos lógicos: *VirarParaDireita, VirarParaEsquerda, IrParaFrente, Pegar, Atirar, Escalar*

Ações do agente

- *Tell*(*KB*,*Percebe*([*Cheiro, Brisa, Brilho, Nada, Nada*],5)
- Função `Make-Action-Query`:

  ASK(*KB*, ∃ *a MelhorAção*(*a*, 5))

Que deveria devolver uma lista de possíveis instâncias de variáveis (*variable bindings*), por exemplo: {*a/Atire, a/ParaFrente*}

O agente então *Tell* para registrar na KB qual ação foi tomada (*o agente não percebe a sua própria ação, por isso tem que lembrá-la!)*

Agente reativo

Possui sentenças que associam diretamente percepções às ações. Exemplo:

∀ *s, b, u, c, t Percebe*([*s, b, Brilho, u, c*], t) => *MelhorAção(Pegar,t)*

que podem ser consideradas *reflexos* ou *instintos*

Esse processo pode ser por meio de regras que fazem uma abstração da percepção imediata para uma interpretação de forma mais útil, p. ex.:

∀ *b, u, g, c, t Percebe*([*Cheiro, b, g, u, c*], t) => *Cheira(t)*

∀ *s, u, g, c, t Percebe*([*s, Brisa, g, u, c*], t) => *Brisa (t)*

∀ *s, b, u, c, t Percebe*([*s, b, Brilho, u, c*], t) => *Em-ouro(t)*

– Assim a conexão entre percepção e ação é feita da seguinte maneira:

∀*t Em-ouro(t)* => *MelhorAção(Pegar,t)*

Agente Reativo

- Questão:
  – *como o agente pode deduzir onde estão os abismos, o Wumpus e ainda ter certeza que ele fez uma busca completa pelo ouro*?

Agente reativo: problemas

- Um agente *reativo puro* :
  - não sabe se ele ja pegou o ouro
  - não sabe se ele está no quadrado de entrada/saída
  - não sabe quando ele deve escalar para a saída ou se a meta já foi satisfeito
  - não sabe se ele decidiu que uma posição é perigosa
- Não pode evitar loops infinitos (pode voltar para um quadrado e receber as mesmas percepções e fazer o mesmo que fez anteriormente)

Agente-KB

O agente guarda todas as suas percepções na KB e raciocina
sobre os axiomas da KB

É possível escrever sentenças (ou regras) que representam percepções correntes e percepções passadas para estender a capacidade do agente reativo

Problema: dificuldade em escrever sentenças que relacionem todas as percepções do passado com o presente

Solução possível: adicionar somente fatos que "importam" para a tomada de decisão do agente, ou seja, manter um modelo interno do mundo atualizado!!!

Axiomas do ambiente

- Definição de adjacência:

$\forall$ ***x*, *y*, *a*, *b* *Adjacente*([*x*, *y*], [*a*, *b*])** $\Leftrightarrow$

$$[a,b] \in \{[x+1,y],[x-1,y],[x,y+1],[x,y-1]\}.$$

- Para representar que a posição do agente muda com o tempo usamos o predicado *Em*(*Agente*,*s*,*t*) ➔ o agente está no quadrado *s* no instante *t*
- O agente pode deduzir propriedades do quadrado a partir de sua percepção atual

$$\forall\, s, t\ Em(Agente, s, t) \wedge Brisa(t) \rightarrow Arejado(s)$$

Axiomas do ambiente

Regras de diagnóstico: relacionam observações a causas

– *Se um quadrado é arejado, algum quadrado adjacente deve conter um poço*

$$\forall\ s\ \mathit{Arejado}(s) \rightarrow \exists\ r\ \mathrm{Adjacente}(r,s) \wedge \mathrm{Poço}(r).$$

– *Se um quadrado não é arejado, nenhum quadrado adjacente contém um poço*

$$\forall\ s\ \neg \mathit{Arejado}(s) \rightarrow \neg\ \exists\ r\ \mathrm{Adjacente}(r,s) \wedge \mathrm{Poço}(r).$$

– *Combinando os dois axiomas acima temos:*

$$\forall\ s\ \mathit{Arejado}(s) \Leftrightarrow \exists\ r\ \mathrm{Adjacente}(r,s) \wedge \mathrm{Poço}(r).$$

Axiomas do ambiente

Regras causais: propriedades causam observações

– *Um poço faz com que todos os quadrados adjacentes fiquem arejados*

$\forall\, r\ \textit{Poço}(r) \rightarrow (\forall\, r\ \textit{Adjacente}(r,s) \rightarrow \textit{Arejado}(s)).$

– *Se todos os quadrados adjacentes a um dado quadrado forem quadrados sem poços, o quadrado não será arejado*

$\forall\, s\ (\forall\, r\ \textit{Adjacente}(r,s) \rightarrow \neg\, \textit{Poço}(r)) \rightarrow \neg\, \textit{Arejado}(s).$

Tipos de raciocínio

- Sistemas que raciocinam com regras causais são também chamados de **Sistemas Baseados em Modelo** (*Model Based Reasoning Systems*)
- Sistemas que raciocinam com regras de diagnóstico são também chamados de **Sistemas de Classificação Heurística**

Agente-KB: escolha da melhor ação

- Exemplo de fato que importa para a tomada de decisão e que não está diretamente ligado a percepção: ***Segura*(*Ouro*,*t*)**, ou seja, um resultado da ação ***Pega***
- Agora podemos construir um agente reativo com estado interno: *o agente já está segurando o ouro?*

$$\forall t\ \textit{Em-ouro}(t) \land \neg\, \textit{Segura}(\textit{Ouro},t) \Rightarrow \textit{MelhorAção}(\textit{Pega},t)$$

***Segura* (*Ouro*,*t*)** não pode ser observado => guardar histórico das mudanças do mundo é essencial

Questão: como representar mudanças do mundo ?

Representação de ações

- Representando pré-condições (seleção da ação executável):

$\forall t$ ***Em-ouro(t) => MelhorAção(Pega,t)***

- Representando efeitos para seleção da ação que atinge metas (raciocínio sobre tempo)

$\forall t$ ***MelhorAção(Pega,t) => Segura(Ouro,t+1)***

Situações

- Representando efeitos para seleção da ação que atinge metas (raciocínio sobre situações)

$\forall$ ***s Em-ouro(s)*** $\Rightarrow$ ***Segura(Ouro, Resultado(Pega,s))***

$\forall$ ***a,x,s Segura(x,s)*** $\wedge$ ***(a*** $\neq$ ***Solta)*** $\Rightarrow$ ***Segura(x, Resultado(a,s))***

Modelo interno do mundo X Raciocínio sobre um histórico de percepções

- Pode ser mostrado que qualquer sistema que toma decisões com base em percepções passadas (*regras diacrônicas*) pode ser re-projetado para usar um conjunto de sentenças sobre o estado presente do mundo (uma vez que essas sentenças sejam atualizadas a cada nova perceção e a cada nova ação realizada pelo agente)

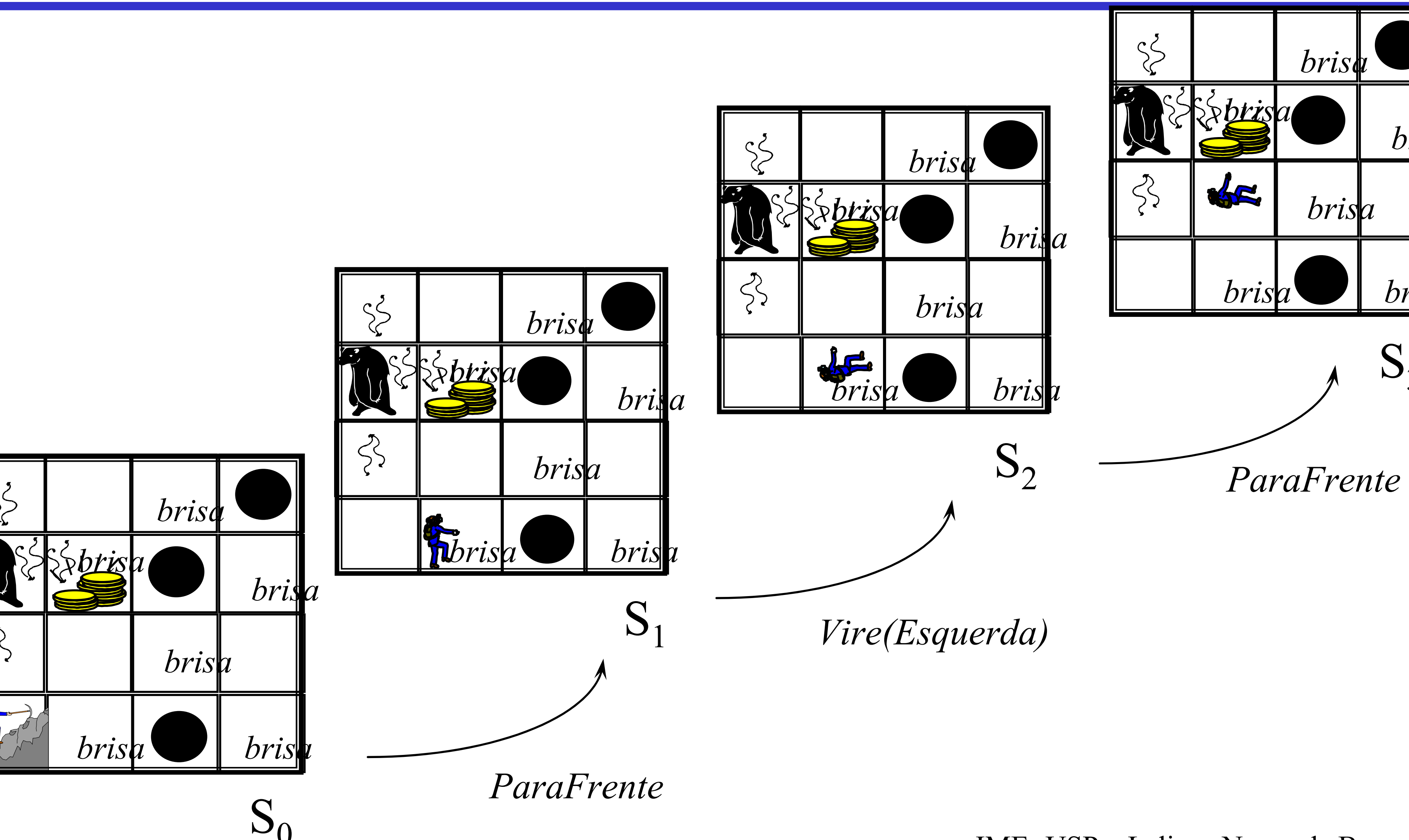
Fazer mudanças na KB

- Maneira mais simples de tratar mudanças no mundo: retirar sentenças (agente está em [1,1]) e substituí-la por outras que atualizam o estado do mundo (agente está em [2,1])
- Agente só poderá responder sobre a situação mais recentemente atualizada (o histórico completo do passado é perdido)

Busca em estados passados e futuros

- Raciocinar sobre situações hipotéticas (possíveis mundos futuros)
- Planejar sobre o futuro

Cálculo de Situações



Cálculo de Situações

- Como representar as mudanças de mundo ?
- através da função ***Resulta(ação, situação)*** que retorna a situação que resulta da execução da ação em alguma dada situação:

$$\boldsymbol{Resultado(ParaFrente, S_0) = S_1}$$

$$\boldsymbol{Resultado(Vire(Esquerda), S_1) = S_2}$$

$$\boldsymbol{Resultado(ParaFrente, S_2) = S_3}$$

Cálculo de Situações

- Uma das maneira de se representar mudanças no mundo em FOL
- Situações são geradas (ligadas) a partir de situações anteriores através de ações.
- Toda relação ou objeto que muda no mundo deve possuir um argumento extra:

Localizado-em(Agente, [1,1], S_0) ^ Localizado-em(Agente, [1,1], S_1)

que descreve a localização do agente nas duas situações iniciais

- relações ou propriedades que não mudam com o tempo não precisam do argumento extra de situação. Ex.: *Tem-parede(0,1)*

Como representar as mudanças de mundo ? → **Axiomas de efeito**

Ações são descritas através de seus efeitos, ou seja, propriedades das situações que resultam com a execução da ação

Ex.: "uma vez pego o ouro ele está onde o agente está"

$$\forall\, s\ \mathit{Em\text{-}ouro}(s) \Rightarrow \mathit{Segura}(\mathit{Ouro}, \underbrace{\mathit{Resultado}(\mathit{Pega},s)}_{s'})$$

Axiomas de Frame

Como descrever as propriedades que continuam valendo depois de certas ações serem executadas?

Exemplo:

"O agente não segura nada depois de uma ação solta":

$$\forall x,s \;\; \neg \mathit{Segura}(x, \mathit{Resultado}(\mathit{Solta},s))$$

"Se o agente segura qualquer coisa e não a solta, então ele a segura no próximo estado"

$$\forall a,x,s \; \mathit{Segura}(x,s) \land (a \neq \mathit{Solta}) \Rightarrow \mathit{Segura}(x, \mathit{Resultado}(a,s))$$

Axioma de sucessor-de-estado

- Maneira mais elegante de se representar os dois axiomas (cada axioma é sobre um predicado e não sobre uma ação):

O que é verdade no mundo $\Leftrightarrow$ [*uma ação tornou verdadeiro* $\vee$ *já era verdade e nenhuma ação tornou falso*] ou

P é verdade $\Leftrightarrow$ [*uma ação tornou P verdade* $\vee$ *P já era verdade e nenhuma ação tornou P falso*]

Exemplo:

$$\forall a,x,s\ Segura(x, Resultado(a,s)) \Leftrightarrow (a = Pega \wedge (Presente(x,s)) \vee ((Segura(x,s) \wedge (a \neq Solta))$$

Frame Problem

Representational frame problem (resolvido com o axioma de sucessores de estado: evita o crescente número de axiomas de frame)

Qualification problem**:** descrições do mundo real requer uma profundidade ilimitada: e se o ouro for escorregadio ou colado no chão ou ... (problema da representação de pré-condições de ações)

Ramification problem: ações reais possuem muitas consequências secundárias (problema da representação de efeitos de ações)

Mundo de Wumpus

- Questão:
  – *como o agente pode deduzir onde estão os abismos, o Wumpus e ainda ter certeza que ele fez uma busca completa pelo ouro*?

Mundo de Wumpus

- Localização é importante. Não pode ser <u>percebida diretamente</u> mas o agente precisa se lembrar:
  – onde esteve
  – o que viu
- para:
  – poder deduzir onde estão os abismos, o Wumpus e ainda para ter certeza de ter feito uma busca completa pelo ouro.
- Localização inicial: $Em(Agente, [1,1],S_0)$

Mais sobre o mundo de wumpus

O que o agente precisa saber:

- *Orientação(Agente,S_0) = Leste*
- Mapa de localizações: valores da função
- Identificar o quadrado da frente
- Identificar o quadrado adjacente
- Localização das paredes
- Controle da orientação

Raciocinando no Mundo de Wumpus

- Localização pode ser *Brisa(l), Cheiro(l)*

  $\forall$ ***l,s Localizado-em(Agent,l,s)*** $\wedge$ ***Brisa(s)*** $\Rightarrow$ ***Brisa(l)***

- Dois tipos de regras que relacionam propriedades do mundo do mesmo estado:
  - Regras causais (propriedades do mundo causam a geração de percepções)

$\forall$ $\boldsymbol{l_1, l_2, s}$ ***Localizado-em(Abismo,*** $\boldsymbol{l_1}$ ***,s)*** $\wedge$ ***Adjacente(***$\boldsymbol{l_1, l_2}$***)*** $\Rightarrow$ ***Brisa(***$\boldsymbol{l_2}$***)***

  - Regras de diagnóstico (a partir da percepção infiro as propriedades do mundo)

$\forall$ ***l,s Localizado-em(Agent,l,s)*** $\wedge$ ***Brisa(s)*** $\Rightarrow$ ***Brisa(l)***

$\forall$ $\boldsymbol{l_1, s}$ ***Cheiro(***$\boldsymbol{l_1}$***)*** $\Rightarrow$ ***(***$\exists$ $\boldsymbol{l_2}$ ***Localizado-em(Wumpus,***$\boldsymbol{l_2}$ ***,s)*** $\wedge$ ***(***$\boldsymbol{l_2 = l_1}$ ***v Adjacente(***$\boldsymbol{l_1, l_2}$***))***

Tipos de raciocínio

- Sistemas que raciocinam com regras causais são também chamados de **Sistemas Baseados em Modelo** (*Model Based Reasoning Systems*)
- Sistemas que raciocinam com regras de diagnóstico são também chamados de **Sistemas de Classificação Heurística**

Caminho de volta

- As sentenças que descrevem as ações são suficientes para garantir que o agente encontre o ouro de maneira segura
- Uma vez que o ouro foi encontrado, o agente deve retornar à saída o mais "rápido" possível

$\forall$ ***s Segura(Ouro,s)*** $\Rightarrow$ ***Localização-goal([1,1], s)***

O agente deve gerar uma sequência de ações para atingir o goal este objetivo => o agente deve planejar

Fazendo planos

- Condição inicial em KB:

  Localizado-em(Agent,[1,1],S_0)

  Localizado-em(Ouro,[1,2],S_0)

  Pergunta: $ASK(KB, \exists s$ Segura($Ouro$, s))

  “*Em que situação eu seguro o Ouro?*”

  Resposta:

  {s/*Resultado(Pega,Resultado(ParaFrente,S_0))*}

  “Vou para frente e então pego o ouro”

Planejamento

Representação de planos como uma sequência de ações $[a_1, a_2, \ldots, a_n]$ (aplicação sucessiva de funções)

ResultadoPlano(p,s) é a situação resultante da execução de *p* em *s*

Pergunta: $ASK(KB, \exists\, p$ Segura$(Ouro, ResultadoPlano(p,S_0))$

Solução: $\{p/[ParaFrente, Pega]\}$

Definição de *ResultadoPlano* em termos de *Resultado*:

$\forall\, s\ \boldsymbol{ResultadoPlano}([\,],s) = s$

$\forall\, a,p,s\ \boldsymbol{ResultadoPlano}([a|p],s) = \boldsymbol{ResultadoPlano}(p,\boldsymbol{Resultado}(a,s))$

Sistemas de Planejamento

- Sistemas de planejamento são processos de raciocínio de *propósito específico* (algoritmos) designados a fazer esse tipo de inferência de maneira *mais eficiente* do que os processos de raciocicínio de propósito geral como a LPO
- Questão: é possível construir um planejador baseado em lógica que seja tão *eficiente* quanto o algoritmo de planejamento POP?