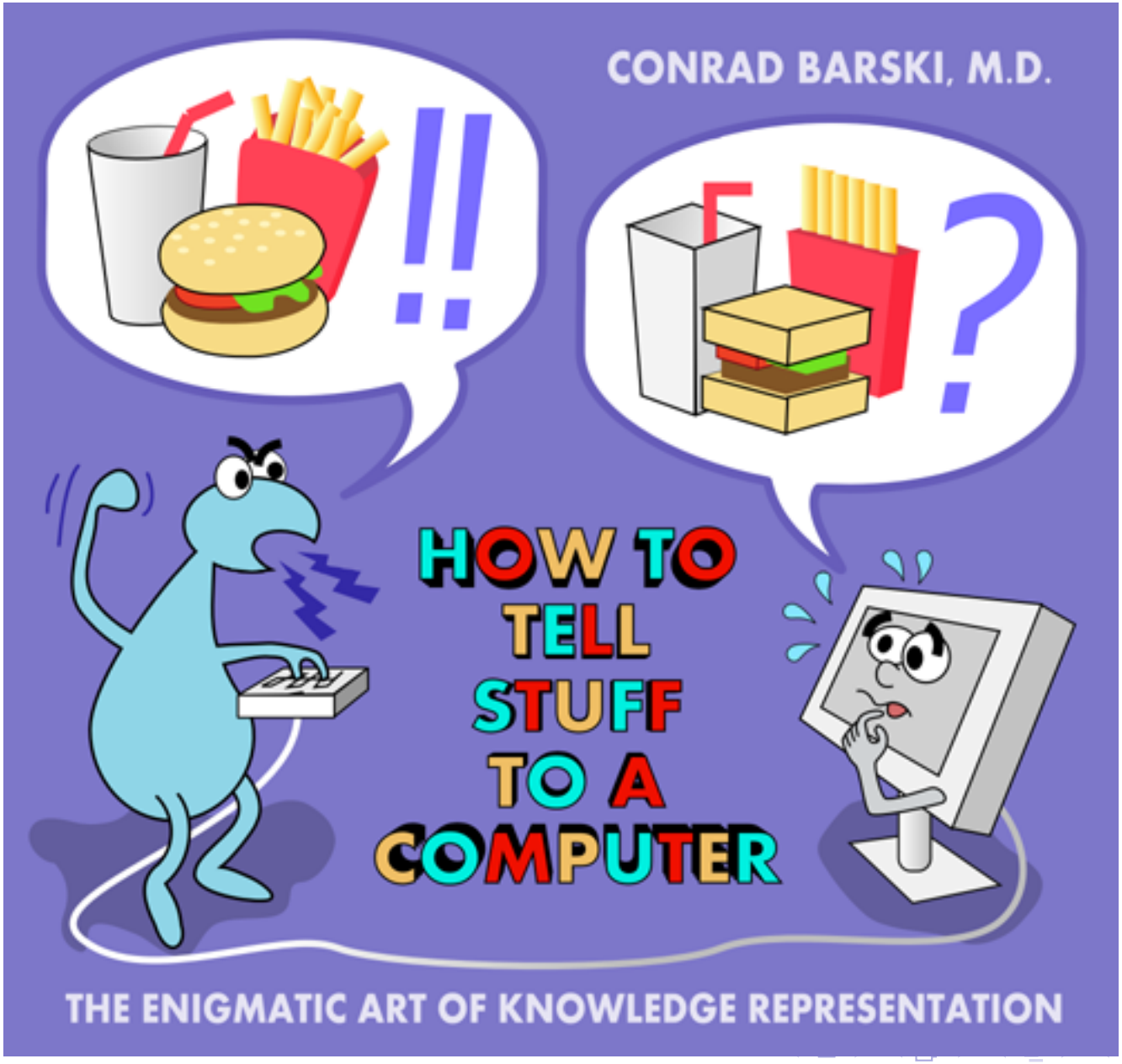
Representação do Conhecimento

May 8, 2016

Representação de Dados e de Conhecimento



Representação de Dados e de Conhecimento

Refs para este tópico:

- caps. 7, 8 e 12, Artificial Intelligence: a Modern Approach, 3rd ed., by Stuart Russell and Peter Norvig
- part II, Artificial Intelligence, 2nd ed., by Elaine Rich and Kevin Knight
- What is a Knowledge Representation?: `groups.csail.mit.edu/medg/ftp/psz/k-rep.html`
- Informal intro to Knowledge Representation and state-of-the-art: `http://lisperati.com/tellstuff/index.html`
- The CycL Language: `http://www.opencyc.org/doc`
- Prolog: `http://www.dcc.fc.up.pt/~vsc/Yap/`

Conhecimento Certo

- **positivo:** "Indivíduos que ressonam têm apneia obstrutiva do sono (OSA)"
- **negativo**: "Indivíduos que não ressonam não têm apneia obstrutiva do sono (OSA)"
- **desconhecido**: "Indivíduos que ressonam podem ou não ter apneia obstrutiva do sono (OSA)"

Conhecimento Incerto

- **positivo:** "Indivíduos que ressonam têm 70% de probabilidade de ter apneia obstrutiva do sono (OSA)"
- **negativo**: "Indivíduos que não ressonam têm 70% de probabilidade de não ter apneia obstrutiva do sono (OSA)"
- **desconhecido**: "Indivíduos portugueses têm 3% de probabilidade de ter apneia obstrutiva do sono (OSA)" (prevalência)

Representação

- Conhecimento x Dados?
- **Conhecimento:** "representação simbólica de aspectos de algum universo de discurso"

Exemplos de "conhecimento"

- José é um funcionário da UP
- Todos os funcionários da UP têm salários maiores que 25.000 euros (:-)
- Todos os funcionários da UP sabem que devem ter um bom estilo de vida
- José não acha que tem um bom estilo de vida
- Todos que sabem que ele deveria ter um bom estilo de vida, mas pensa que não tem, estão desapontados

Representação

- **Dados:** "representação simbólica de aspectos **simples** de algum universo de discurso"
- **Dado**: caso especial de "conhecimento"

Exemplos de "dados"

- José é casado com Maria
- José é funcionário da UP
- O salário médio da UP é de 25.000 euros

Representação

- Representação do Conhecimento: expressar **conhecimento** de forma tratável pelo computador.

Diferentes formalismos

Linguagem natural	Regras
Bases de dados	Árvores de decisão
Frames	Lógica
Scripts	Ontologias
Redes Semânticas	Redes causais
Algoritmos genéticos	Redes neuronais
Restrições	Orientação a objetos
Linguagens	etc!

Representação em Linguagem Natural

Texto Clínico

"Enviada por densidade assimétrica no QSE da mama esquerda. Esta alteração existe desde 2005 mas a avaliação ecográfica do exterior sugere a necessidade de biópsia. Exame mamário com alteração palpável com cerca de 30 mm no QSE da mama esquerda."

Desvantagens:

ambígua, redundante, pouca estrutura, sintaxe e semântica não são bem entendidas.

Representação em Bases de Dados

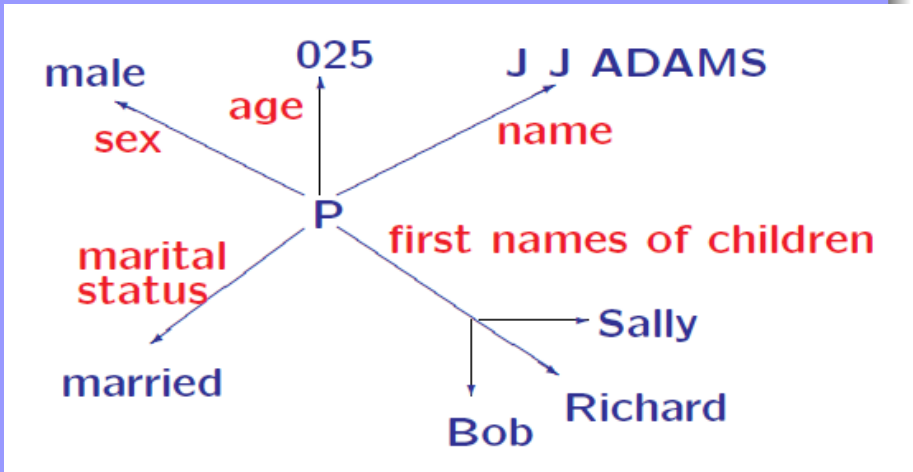
Base de Dados

```
person
record = { name : max 20 characters
           age  : 3 digits in range 000-120
           sex  : male or female
           marital status : married, bachelor,
                            spinster, divorced,
                            widowed, or engaged
           first names of children : up to 10 names
                            each max 15 characters
         }
```

Representação em Bases de Dados: uma instância

Instância

J. J. ADAMS
025
male
married
Sally
Richard
Bob



Discussão

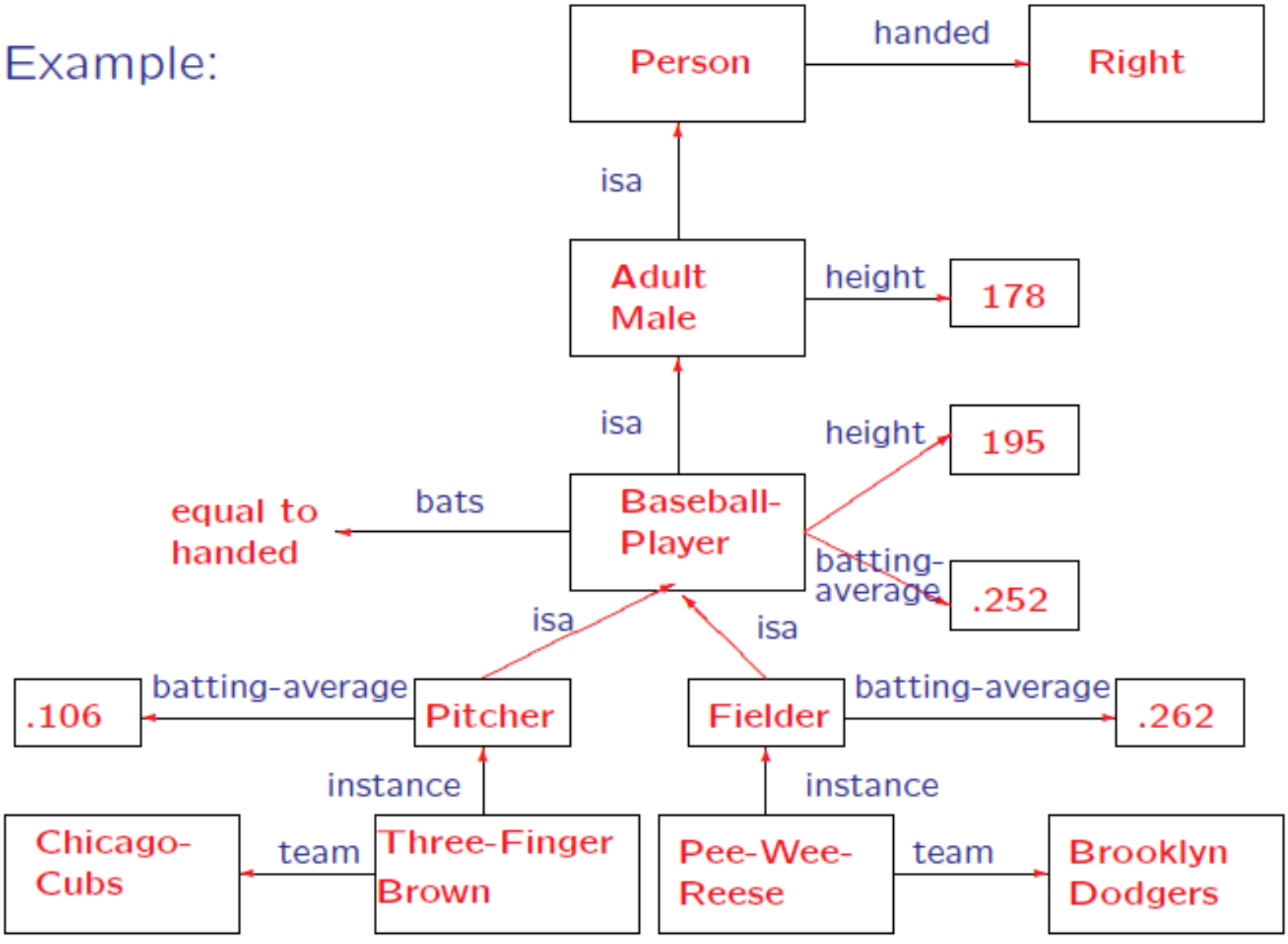
- apenas aspectos simples podem ser representados (dados)
- entidades e relações
- Reasoning = lookup

Representação em uma única tabela

Usual: agregar dados em uma única tabela!

Patient	Location	Size	Date	Calcifications
P1	C	0.1	20050403	F, A
P1	C	0.2	20060412	F
P1	9	0.1	20060412	A
P2	12	0.3	20050415	M
...	...	...	...	...

Representação em Redes Semânticas



Propriedades de Redes Semânticas

- permite estruturar o conhecimento para refletir a parte do universo que está sendo representada
- valores "default"
- sintaxe clara, mas semântica precisa ser trabalhada

Exemplo baseado em CycL (linguagem)

"Bill Clinton belongs to the collection of U.S. presidents"

```
(#$isa #$BillClinton #$UnitedStatesPresident)
```

"All trees are plants"

```
(#$genls #$Tree-ThePlant #$Plant)
```

"Paris is the capital of France."

```
(#$capitalCity  #$France  #$Paris)
```

"if OBJ is an instance of the collection SUBSET and SUBSET is a subcollection of SUPERSET, then OBJ is an instance of the collection SUPERSET".

```
(#$implies
   (#$and
     (#$isa ?OBJ ?SUBSET)
     (#$genls ?SUBSET ?SUPERSET))
   (#$isa ?OBJ ?SUPERSET))
```

Frames

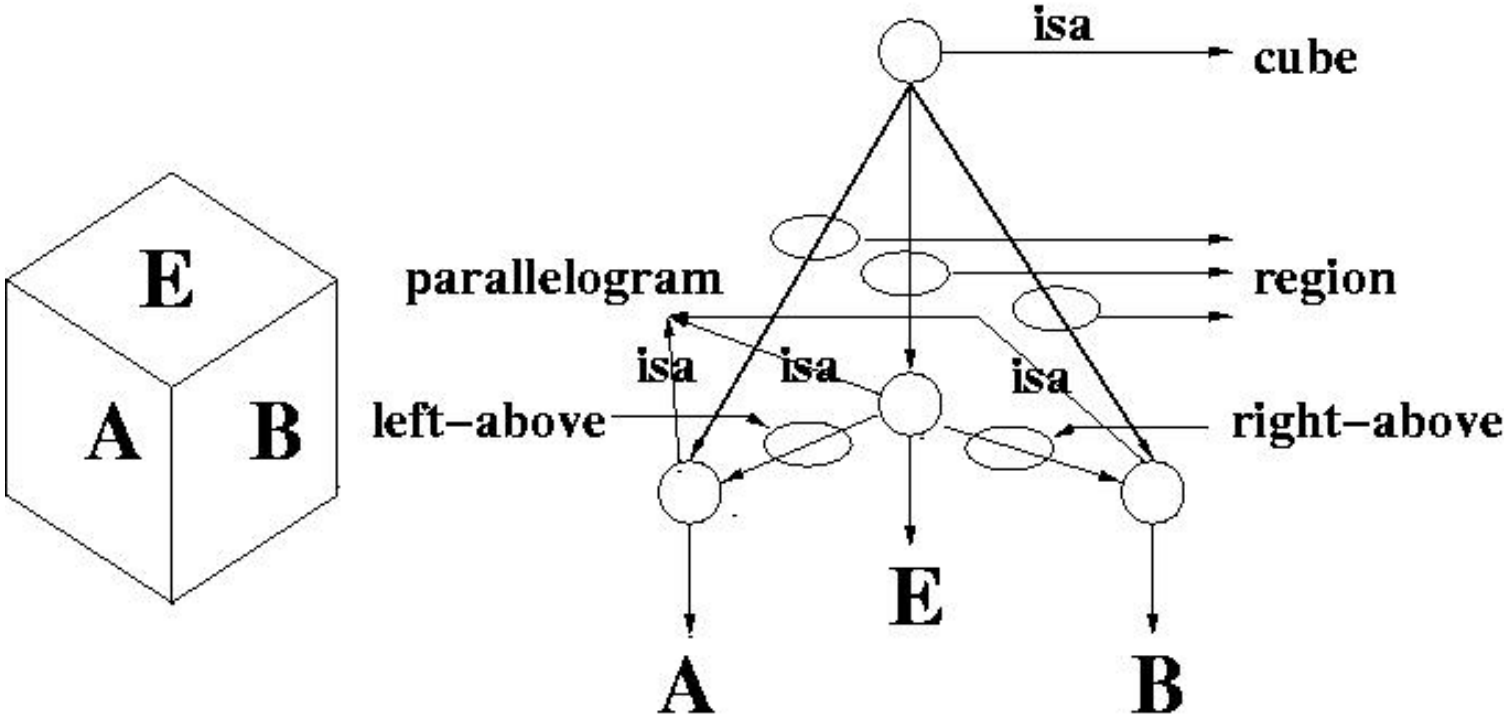
- Um "frame" consiste numa coleção de "slots", cujo conteúdo pode ser um valor ou um apontador para outro Frame.

Festa de aniversário	
Vestuário:	social desportivo
Presente:	deve agradar o aniversariante deve ser comprado e embrulhado
Jogos:	escondidas colocar o rabo no burro
Decoração:	balões, brindes, papel crepe
Menu:	Bolo, Gelado, Refri, Cachorro quente
Bolo:	acender velinhas, assoprar velinhas, fazer um pedido, cantar parabéns
Gelado:	napolitano

Frames

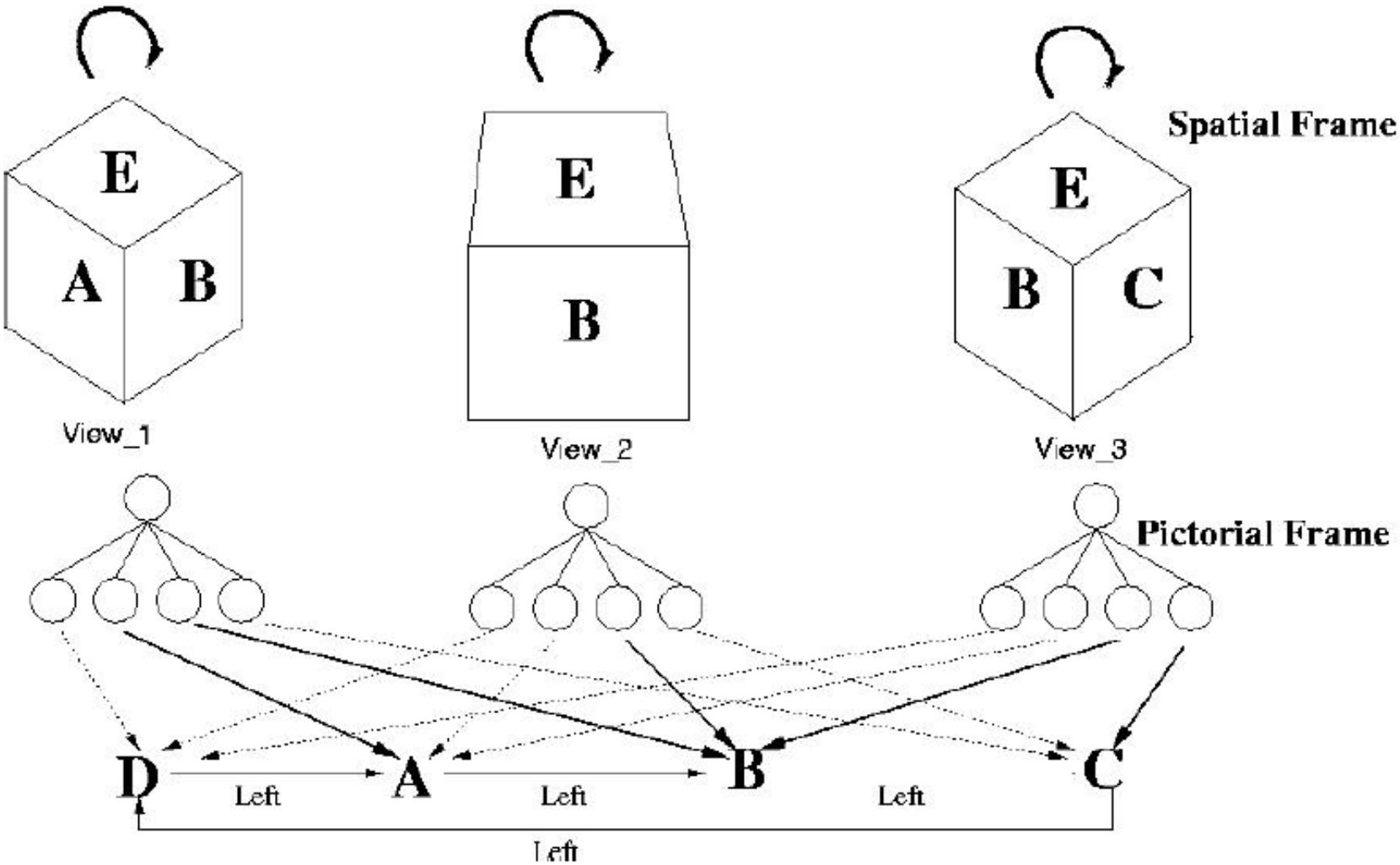
- Um "frame" consiste numa coleção de questões a serem respondidas sobre uma situação hipotética: especifica as questões e os métodos.
  - O que causou (agente)?
  - Qual é o propósito (intenção)?
  - Quais são as consequências (efeitos)?
  - A quem afeta (receptor)?
  - Como é feito (instrumentos/métodos)?

Exemplo



Objeto composto por relações.

Exemplo



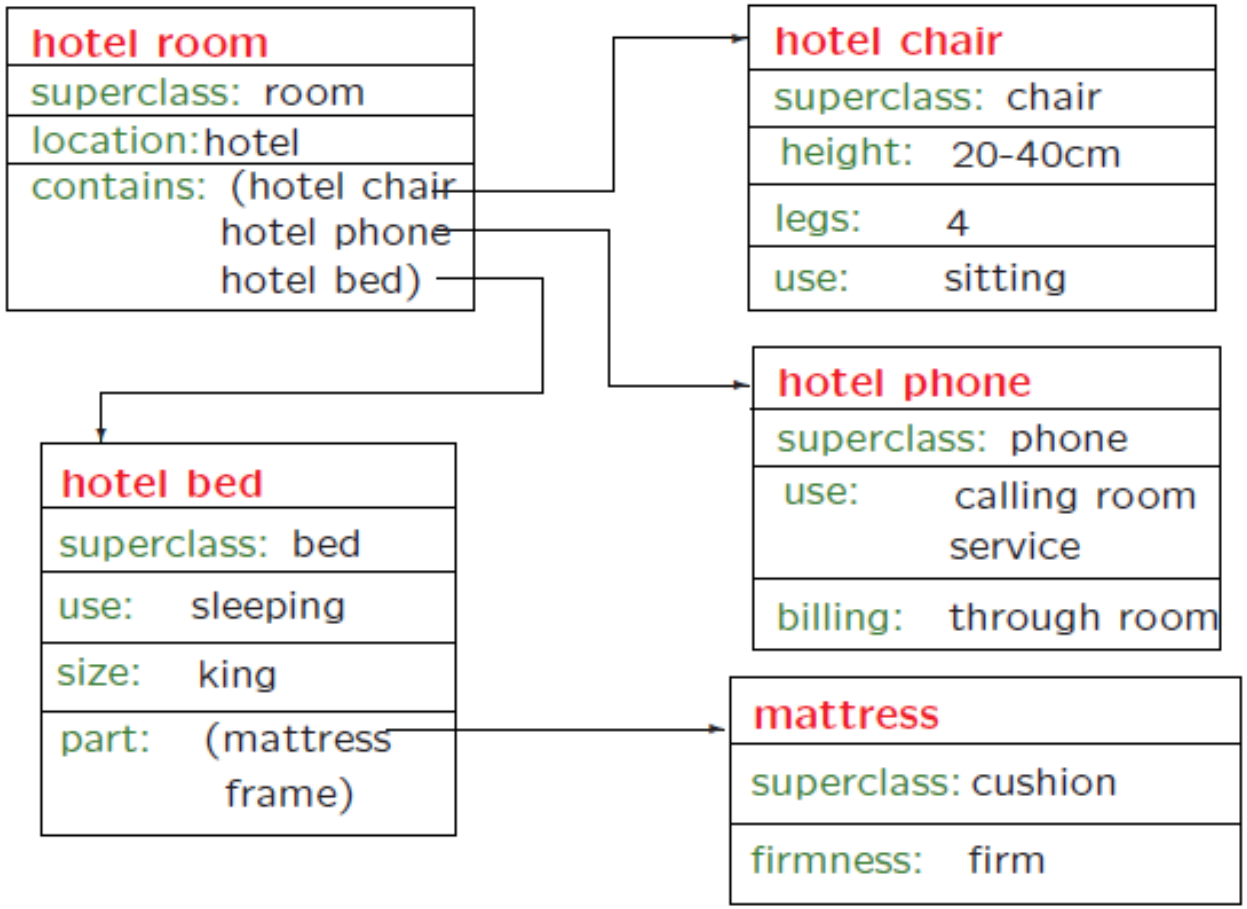
Diferentes aspectos de um cubo.

Frame para um aspecto do cubo

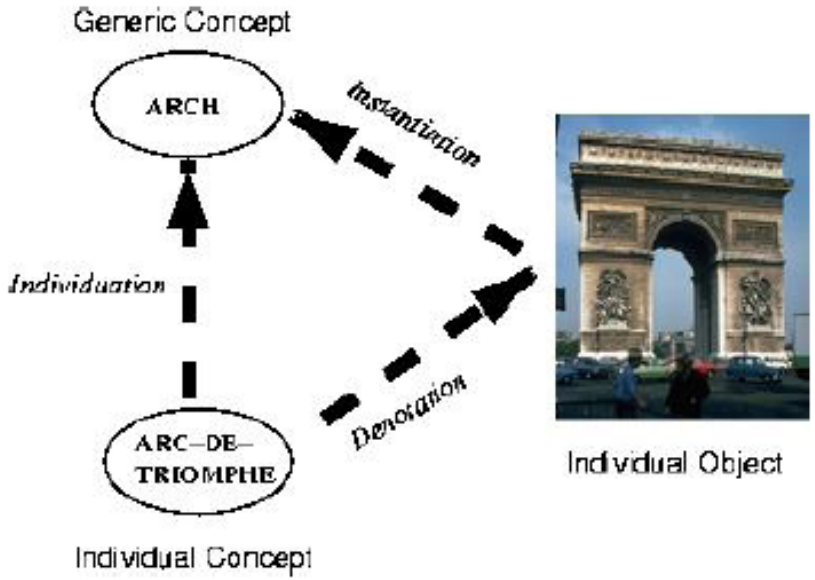
Um aspecto de um cubo usando representação em Frame

View-of-a-Cube		
Slot	Filler	Constraint
Name	View_1	
region_of	A	parallelogram & visible
region_of	B	parallelogram & visible
region_of	C	parallelogram & invisible
region_of	D	parallelogram & invisible
region_of	E	parallelogram(E) & visible & left-above(E,A) & right-above(E,B)

Mais um exemplo



Importante!



- distinguir:
  - conceitos (representações) e objetos (instâncias)
  - conceitos individuais e conceitos gerais

Associação de procedimentos à representação

Procedimentos

rectangle	
superclass:	polygon
Coordinates:	(0cm,0cm)
length:	5cm
width:	2cm
area:	procedure(z) length(z) * width(z)
perimeter:	procedure(z) 2 * (length(z) + width(z))

Scripts

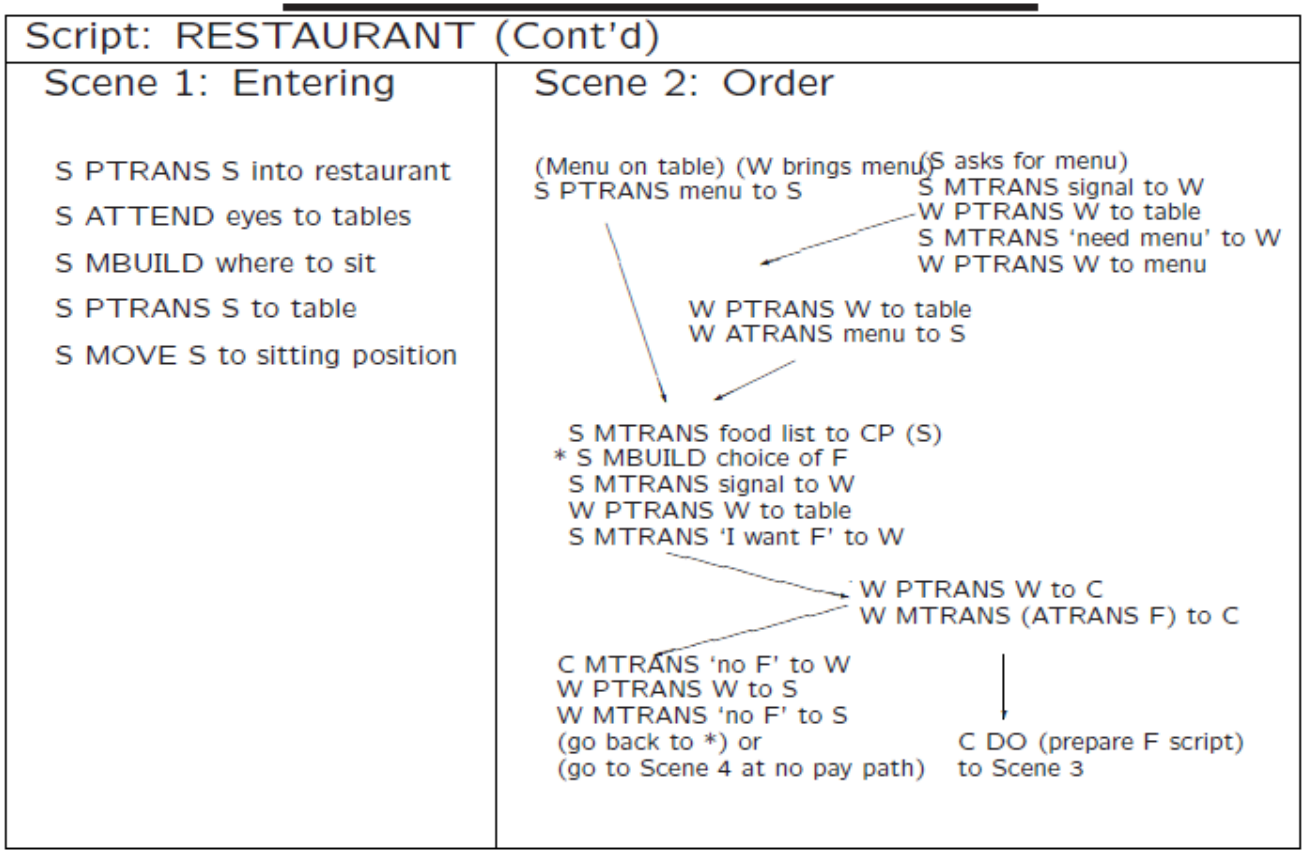
- Um "script" é uma representação estruturada que descreve uma sequência de eventos em um determinado contexto.
  - Estende os "frames" através de representação explícita de ações e mudanças de estados.
  - Define primitivas para descrever o universo:
    - PTRANS transferência física de um objeto ("go")
    - ATRANS transferência de relações ("give")
    - MTRANS transferência mental ("tell")

Exemplo de Script

Script para um restaurante

Script: RESTAURANT			
Track:	Coffee Shop	Entry cond.:	S hungry
Props:	Tables		S has money
	Menu		
	F=Food	Results:	S has less money
	Check		O has more money
	Money		S is not hungry
Roles:	S=Customer		
	W=Waiter		
	C=Cook		
	M=Cashier		
	O=Owner		

Script para um restaurante (cont.)



Representação procedural ou declarativa

- Como? Procedural
- O que? Declarativo

Propriedades das boas Representações

- objetos importantes e suas relações estão explícitos
- expressam restrições que são naturais
- representam objetos e relações de forma conjunta
- omite detalhes irrelevantes
- transparente: fácil de entender
- completa
- concisa
- de armazenamento e recuperação rápidos
- "computáveis"

Propriedades das boas Representações

- Parte léxica que determina quais símbolos devem ser utilizados
- Parte estrutural que descreve a forma (restrições) como os símbolos podem ser organizados
- Parte procedural que especifica procedimentos de acesso que permitem a criação e modificação de descrições além de permitir fazer perguntas
- Parte semântica que estabelece uma forma de associar "significado" à descrição

Propriedades das boas Representações

Por exemplo, redes semânticas:

- Parte léxica: nodes, links, link labels
- Parte estrutural: grafo dirigido, com arestas etiquetadas
- Parte procedural: constructors, readers, writers, erasers (para criar e modificar o grafo)
- Parte semântica: significado dos nós e arestas depende da aplicação

Representação Lógica

- Linguagens:
  - **sintaxe**: descreve as possíveis configurações da linguagem que constituem sentenças válidas.
  - **semântica**: determina o significado de cada sentença.
- exemplo: $x > y$,
  - sintaxe: se x é um número e y é um número, então $x > y$ é uma sentença sobre números.
  - semântica: se $x > y$ retorna verdadeiro, senão retorna falso.

Representação Lógica

- Linguagem com sintaxe e semântica precisas: **lógica**.
- **Mecanismo de inferência**: derivado da sintaxe e da semântica.
- Importante: distinguir entre os fatos e sua representação
  - não podemos colocar todos os fatos do mundo no computador!
  - neste caso, devemos operar em representações dos fatos (codificação em alguma linguagem)
- **Raciocínio**: processo de construir novas configurações a partir de configurações já existentes.
- Bom raciocínio deve assegurar que as novas configurações representam fatos que se seguem dos fatos já existentes (lógica monotônica).

Representação Lógica

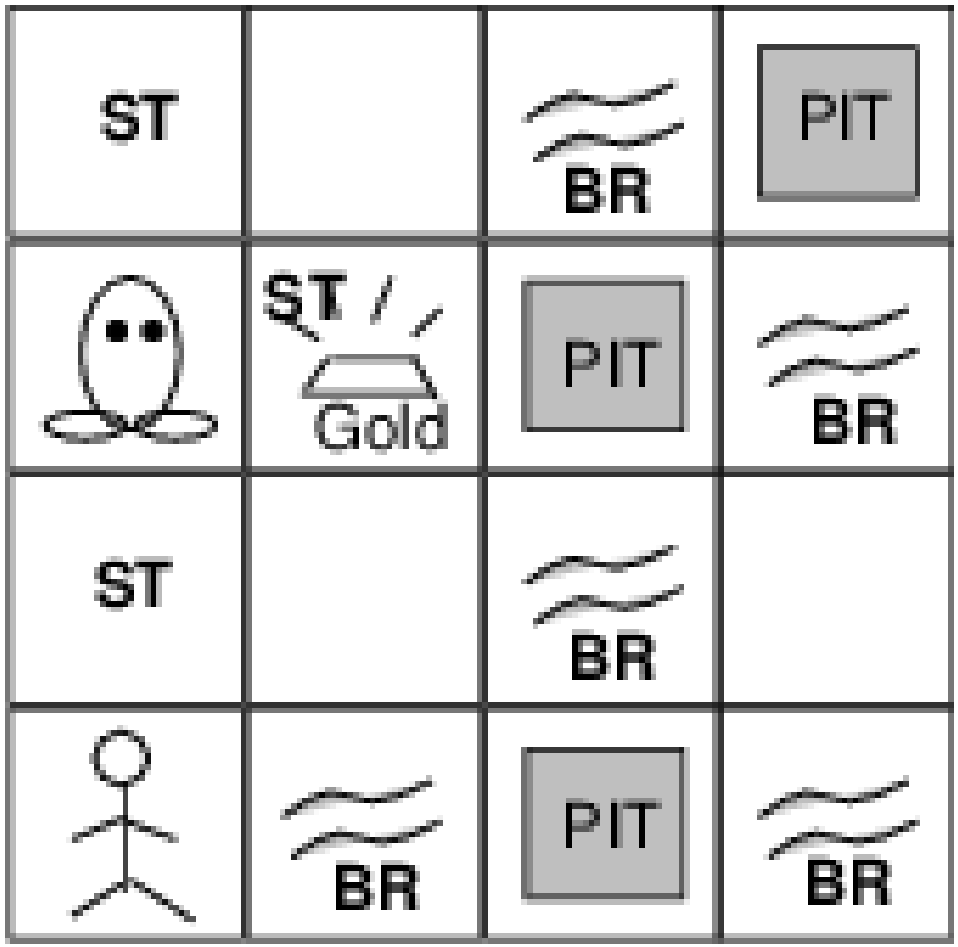
- **"Entailment"**: relação entre sentenças tal que novas sentenças geradas são verdadeiras, dado que as anteriores também são.
- $KB \models \alpha$ (consequência lógica)
- Mecanismo de inferência:
  - dada uma base de conhecimento KB, pode gerar novas sentenças que seguem de KB.
  - dada uma base de conhecimento e uma sentença α, pode dizer se α é consequência lógica de KB.
  - é **sound** ou **truth-preserving** se somente produzir sentenças que sejam consequência lógica de KB.

Representação Lógica

- Prova: procedimento de inferência "sound".
- Analogia: procurar uma agulha num palheiro.
  - "entailment": a agulha está no palheiro.
  - prova: encontrar a agulha.
  - palheiro de tamanho finito + procedimento sistemático de busca → agulha vai ser encontrada: procedimento de inferência **completo**.
- Como obter um procedimento "sound"?
  - passos de inferência devem respeitar a semântica das sentenças já existentes no KB.
  - derivar novas sentenças que sejam consequência lógica dos fatos já representados no KB.

Representação Lógica

Mundo do Wumpus :-)



Representação Lógica

- **Representação**: duas classes de linguagens, programação e natural.
- Vantagens de linguagens de programação:
  - descrever algoritmos e estruturas de dados concretas.
  - Ex: `World[2,2]` $\leftarrow$ `Pit`.
- Desvantagem: pouca expressividade. Como representar:
  - "há um buraco em [2,2] **ou** [3,1]"?
  - "há um monstro em **algum** quadrado"?

Representação Lógica

- **Lógica proposicional** e **lógica de primeira ordem** (cálculo de predicados de primeira ordem com igualdade).
- Lógica proposicional: símbolos são proposições. Ex: D pode ter a interpretação de que o wumpus está morto. Pode assumir valor falso ou verdadeiro.
  - símbolos proposicionais combinados através de **conectivos** booleanos formando sentenças mais complexas.
  - Linguagem bem simples.
- Lógica de primeira ordem: **objetos** e **predicados** relacionando objetos.
  - Admite **quantificadores** ($\forall$ e $\exists$).
  - Mais expressiva do que proposicional.

Lógica Proposicional

Sintaxe:

S	$\rightarrow$	$AS \mid CS$
AS	$\rightarrow$	**True** $\mid$ **False** $\mid P \mid Q \mid R \mid \ldots$
CS	$\rightarrow$	(S) $\mid$ S C S $\mid \neg$ S
C	$\rightarrow$	$\wedge \mid \vee \mid \Leftrightarrow \mid \Rightarrow$

Lógica Proposicional

Exemplo: $S = ((P \vee H) \wedge \neg H) \Rightarrow P$ é uma fórmula válida.

P	H	$P \vee H$	$(P \vee H) \wedge \neg H$	S
F	F	F	F	T
F	T	T	F	T
T	F	T	T	T
T	T	T	F	T

- P: wumpus está na posição [1,3].
- H: wumpus está na posição [2,2].
- Se sabemos que $(P \vee H)$ é verdadeiro e $\neg H$ também é verdadeiro, então o wumpus só pode estar na posição [1,3].

Lógica Proposicional

- Agente para o mundo do wumpus!
- B: brisa, S: mau cheiro, W: wumpus.
- $\neg S_{1,1}, \neg S_{2,1}, S_{1,2}, \neg B_{1,1}, B_{2,1}, \neg B_{1,2}$: fatos.
- Regras:

  $R_1 : \neg S_{1,1} \Rightarrow \neg W_{1,1} \wedge \neg W_{1,2} \wedge \neg W_{2,1}$

  $R_2 : \neg S_{2,1} \Rightarrow \neg W_{1,1} \wedge \neg W_{2,1} \wedge \neg W_{2,2} \wedge \neg W_{3,1}$

  $R_3 : \neg S_{1,2} \Rightarrow \neg W_{1,1} \wedge \neg W_{1,2} \wedge \neg W_{2,2} \wedge \neg W_{1,3}$

  $R_4 : S_{1,2} \Rightarrow W_{1,1} \vee W_{1,2} \vee W_{1,3} \vee W_{2,2}$

Lógica Proposicional

- Problemas com lógica proposicional:
- muitas proposições para o quadrado 4x4.
- Ex: "não ande para a frente se o wumpus estiver na sua frente" precisa de um conj de 64 regras (16 quadrados x 4 orientações).
- não tem memória do caminho a menos que se represente uma proposição para cada instante no tempo.
- Ex: move para $A_{2,1}$ se torna verdade e $A_{1,1}$ se torna falso. Mas pode ser importante guardar o fato de que o agente esteve em $A_{1,1}$.
- problema: não sabemos o tempo que vai levar para terminar o jogo.

Lógica Proposicional

- Exemplo de proposições adicionais:
  $A^0_{1,1} \wedge East^0_A \wedge W^0_{2,1} \Rightarrow \neg Forward^0$
  $A^1_{1,1} \wedge East^1_A \wedge W^1_{2,1} \Rightarrow \neg Forward^1$
  $A^2_{1,1} \wedge East^2_A \wedge W^2_{2,1} \Rightarrow \neg Forward^2$
  $\vdots$
- índice no topo de cada símbolo indica tempo.
- para 100 unidades de tempo: 6400 destas regras, somente para dizer: "não mova para a frente se o wumpus estiver lá".
- lógica de primeira ordem: reduz as 6400 para apenas 1!

Lógica de Primeira Ordem (First-Order Logic or FOL)

- **objetos** e **relações** entre objetos, **propriedades**, **funções**.
- Objetos: pessoas, casas, números, teorias, Fernando Henrique, cores, jogos de futebol, séculos etc.
- Relações: irmão/irmã de, parte de, maior que, tem cor, ocorreu depois, pertence etc.
- Funções: pai de, melhor amigo de, vencedor de, um mais que etc.
- Ex: "quadrados vizinhos ao quadrado do wumpus têm mau cheiro". Objetos: wumpus, quadrado; Propriedade: mau cheiro; Relação: vizinhança.
- Motivação para o uso de lógica de primeira ordem: formalismo mais estudado e melhor entendido que outras abordagens.

Lógica de Primeira Ordem (First-Order Logic or FOL)

$$
\begin{array}{lcl}
S & \rightarrow & AS \mid SCS \mid QVar, \ldots S \mid \neg S \mid (S) \\
AS & \rightarrow & Pred(Term, \ldots) \mid Term = Term \\
Term & \rightarrow & Func(Term, \ldots) \mid Const \mid Var \\
C & \rightarrow & \Rightarrow \mid \wedge \mid \vee \mid \Leftrightarrow \\
Q & \rightarrow & \forall \mid \exists \\
Const & \rightarrow & A \mid X_1 \mid John \ldots \\
Var & \rightarrow & a \mid x \mid s \mid \ldots \\
Pred & \rightarrow & Mother \mid LeftLegOf \mid \ldots
\end{array}
$$

Lógica de Primeira Ordem (First-Order Logic or FOL)

- Agente lógico para o mundo do wumpus.
- três tipos de agentes: **reflexos**, **baseados em modelo** e **baseados em objetivos**.
- 1º passo: definir a interface com o mundo externo
- sentença (interface) típica:
  `Percept([Maucheiro,Brisa,Brilho,N,N],5)`, onde:
  - elem1: percebe ou não percebe mau cheiro,
  - elem2: percebe ou não percebe brisa,
  - elem3: percebe ou não percebe brilho,
  - elem4: percebe ou não percebe parede,
  - elem5: percebe ou não percebe grito (wumpus sendo morto).
- Ações: Turn(Right), Turn(Left), Forward, Shoot, Grab, Release, Climb.

Lógica de Primeira Ordem (First-Order Logic or FOL)

- Um agente reflexo simples.
- $\forall s, b, u, c, t \; P([s, b, Brilho, u, c], t) \Rightarrow Action(Grab, t)$
- $\forall b, g, u, c, t \; P([MauCheiro, b, g, u, c], t) \Rightarrow MauCheiro(t)$
- $\forall s, g, u, c, t \; P([s, Brisa, g, u, c], t) \Rightarrow Brisa(t)$
- $\forall s, b, u, c, t \; P([s, b, Brilho, u, c], t) \Rightarrow Ouro(t)$
- $\forall t \; AtOuro(t) \Rightarrow Action(Grab, t)$

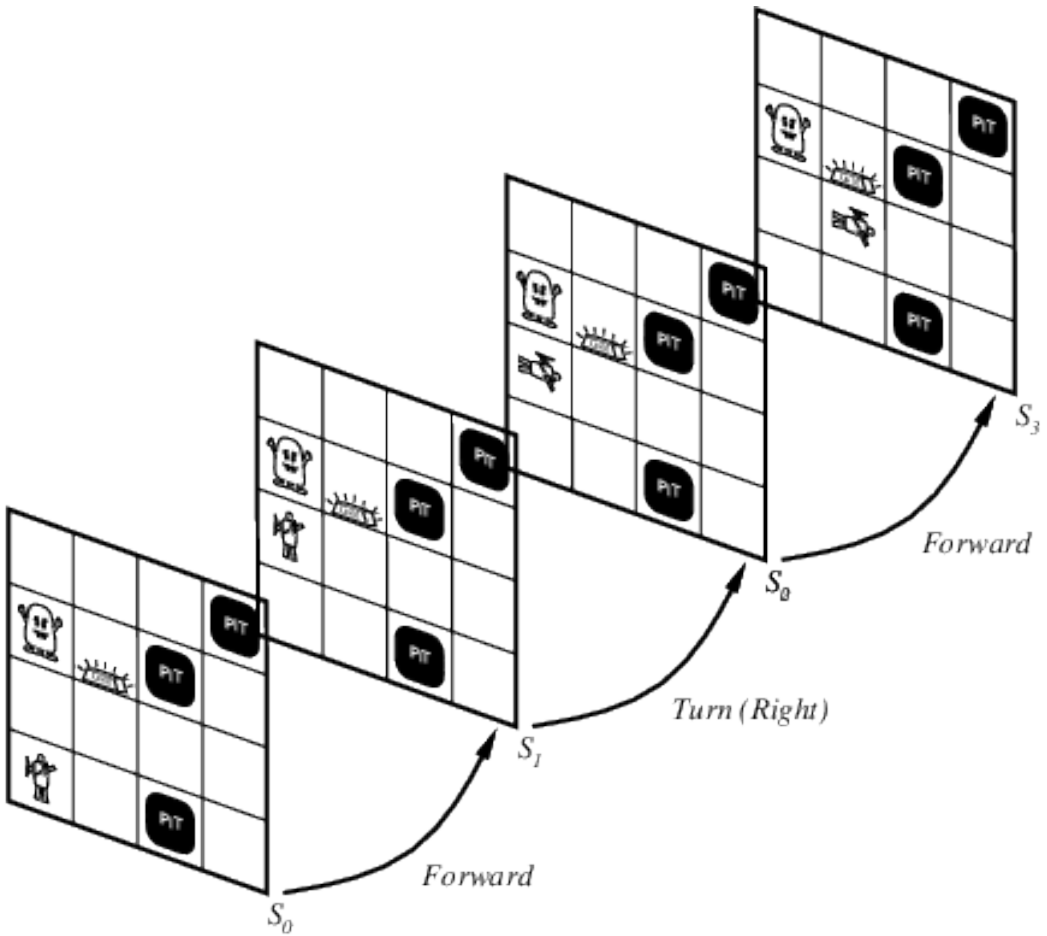
Lógica de Primeira Ordem (First-Order Logic or FOL)

- Limitações de um agente reflexo:
  - não faz parte da percepção deste tipo de agente saber onde está ou se está com o ouro.
  - é incapaz de evitar "loops". Ex: assuma que o agente conseguiu pegar o ouro e está no caminho de volta para casa. Se passar novamente pelo mesmo quadrado visitado na ida, entra em loop.
  - problema: não está representado neste agente o fato dele estar carregando o ouro e a situação ser diferente da situação da ida.
- precisa de *representação de modificações* no mundo.

Lógica de Primeira Ordem (First-Order Logic or FOL)

- **Representação de modificações**: uma das áreas mais importantes em representação do conhecimento.
- regras *diacrônicas*.
- representação de *situações* e *ações* não é diferente de representação de objetos e relações.
- *Cálculo de Situações*: forma de descrever modificações em lógica de primeira ordem.

Lógica de Primeira Ordem (First-Order Logic or FOL)



Lógica de Primeira Ordem (First-Order Logic or FOL)

- Considera o mundo como uma sequência de **situações**.
- formato: At(Agente,posição,situação). Ex: $At(Agent, [1, 1], S_0) \wedge At(Agent, [1, 2], S_1)$
- cálculo de situações utiliza $Result(action, situation)$ para representar a situação decorrente da execução de uma ação em situação anterior.
- Ex:
  - $Result(Forward, S_0) = S_1$
  - $Result(Turn(Right), S_1) = S_2$
  - $Result(Forward, S_2) = S_3$

Lógica de Primeira Ordem (First-Order Logic or FOL)

- Ações: são descritas através de seus efeitos:

Axiomas de efeito

$Portable(Ouro)$
$\forall s\ AtOuro(s) \Rightarrow Present(Ouro, s)$
$\forall x, s\ Present(x, s) \wedge Portable(x) \Rightarrow Holding(x, Result(Grab, s))$
$\forall x, s\ \neg Holding(x, Result(Release, s))$

- não suficiente para saber se o agente está segurando o ouro ou continua segurando o ouro.

Lógica de Primeira Ordem (First-Order Logic or FOL)

- necessário: regras para dizer se o mundo continuou o mesmo.

Axiomas de frame

$\forall a, x, s\ Holding(x, s) \wedge (a \neq Release) \Rightarrow Holding(x, Result(a, s))$
$\forall a, x, s\ \neg Holding(x, s) \wedge (a \neq Grab \vee \neg(Present(x, s) \wedge Portable(x))$
$\Rightarrow \neg Holding(x, Result(a, s))$

- combinação de axiomas de efeito e de frame:
  verdadeiro posteriormente
  $\Leftrightarrow$ [uma ação fez ser verdadeiro $\vee$ já era verdadeiro antes]

Lógica de Primeira Ordem (First-Order Logic or FOL)

Axioma do estado sucessor

$$\forall a, s, x \; Holding(x, Result(a, s)) \Leftrightarrow [$$
$$(a = Grab \wedge Present(x, s) \wedge Portable(x)) \vee$$
$$(Holding(x, s) \wedge a \neq \; Release)$$
$$]$$

Necessário para cada predicado que pode mudar seu valor no decorrer do tempo.

Lógica de Primeira Ordem (First-Order Logic or FOL)

- Dedução de "propriedades escondidas".
  - $\forall l, s\ At(Agent, l, s) \wedge\ Brisa(s) \Rightarrow Fresco(l)$
  - $\forall l, s\ At(Agent, l, s) \wedge\ MauCheiro(s) \Rightarrow MauCheiroso(l)$
- Regras *sincrônicas* para relacionar propriedades de um estado ao mesmo estado.
  - Causais (sistemas baseados em modelos):
    - $\forall l_1, l_2, s\ At(Wumpus, l_1, s) \wedge Adj(l_1, l_2) \Rightarrow MauCheiroso(l_2)$
    - $\forall l_1, l_2, s\ At(Buraco, l_1, s) \wedge Adj(l_1, l_2) \Rightarrow Fresco(l_2)$
  - Diagnósticas (sistemas baseados em diagnósticos):
    - $\forall l, s\ At(Agent, l, s) \wedge\ Brisa(s) \Rightarrow Fresco(l)$
    - $\forall l, s\ At(Agent, l, s) \wedge\ MauCheiro(s) \Rightarrow MauCheiroso(l)$
    - $\forall l_1, s\ MauCheiroso(l_1) \Rightarrow (\exists l_2\ At(Wumpus, l_2, s) \wedge (l_2 = l_1 \vee Adj(l_1, l_2))$

Regras proposicionais

```
if:   Hib.prior = 1 and not Hib_inactive and Hib1_age_in_months >= 12
      and Hib2_final_parameters_met
then: due.Hib2_final

if:   Hib.prior = 1 and not Hib_inactive and Hib1_age_in_months < 12
      and Hib2_parameters_met
then: due.Hib2

if:   Hib.prior = 1 and not Hib_inactive and Hib1_age_in_months >= 12
      and not Hib2_final_parameters_met
then: next.Hib2_final

if:   Hib.prior = 1 and not Hib_inactive and Hib1_age_in_months < 12
      and not Hib2_parameters_met
then: next.Hib2
```

source: Decision Support and Expert Systems in Public Health, in Public Health Informatics and Information Systems, edited by Patrick W. O'Carrol

Regras de primeira ordem

$$\begin{aligned} same\_finding(F_1, F_2) \longleftarrow\ & MLOView(F_1) \wedge CCView(F_2) \wedge \\ & nipple\_distance(F_1, D_1) \wedge nipple\_distance(F_2, D_2) \wedge \\ & \big(abs(D_1 - D_2) < \epsilon\big) \wedge \\ & side(F_1, left) \wedge side(F_2, left) \wedge \\ & quadrant(F_1, upper\_outer) \wedge quadrant(F_2, upper\_outer) \wedge \\ & massShape(F_1, oval) \wedge massShape(F_2, oval). \end{aligned}$$

$$\begin{aligned} previous\_finding(F_1, F_2) \longleftarrow\ & mammo(P, F_1) \wedge mammo(P, F_2) \wedge \\ & date(F_1, D_1) \wedge date(F_2, D_2) \wedge \\ & (D_1 < D_2 \vee D_2 < D_1) \end{aligned}$$

This rule relates two findings F_1 and F_2 for the same patient P, separated in time (date of F_1 is before or after the date of F_2). It can be further used to simulate temporal reasoning in the context of other rules such as:

$$\begin{aligned} is\_malignant(A) \longleftarrow\ & mass(A, present) \wedge previous\_finding(A, B) \wedge \\ & \big(massSize(A) < massSize(B)\big) \wedge calc(B, present) \wedge \\ & previous\_finding(A, C) \wedge calcFineLinear(C, yes) \end{aligned}$$

Árvores de Decisão

- próximo conjunto de slides.

Redes de Bayes

- conjunto de slides após árvores de decisão.

Estado da Arte