

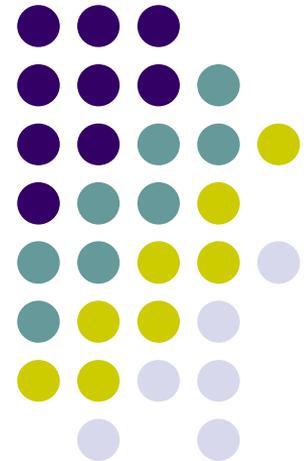
Sistemas de Representação e Raciocínio

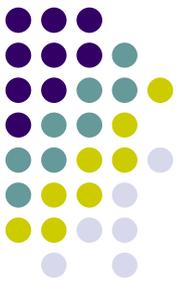
Introdução à Inteligência Artificial

Profa. Josiane

Baseado no material de David Poole,
Alan Mackworth e Randy Goebel

Abril/2007





Inteligência Artificial

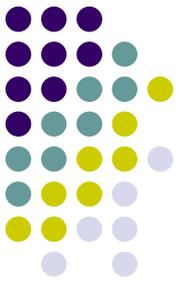
- **Objetivo específico:** Entender os princípios que tornam possível o comportamento inteligente, em sistemas naturais ou artificiais
- **Objetivos de engenharia:** Especificar métodos para projetar artefatos úteis e inteligentes
- Hipótese central da inteligência computacional
 - Sistemas simbólicos: raciocínio é manipulação de símbolos
 - Tese de Church-Turing: qualquer manipulação de símbolos pode ser executada por uma Máquina de Turing



O Conhecimento é importante

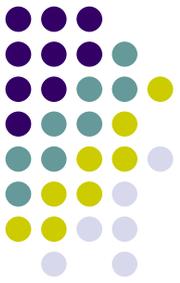
- Executar tarefas que envolvem inteligência requerem conhecimento armazenado
- Conhecimento é a informação sobre algum domínio ou assunto de uma área, ou sobre como fazer alguma coisa
 - Conhecimento de senso comum
- Computadores são bons em tarefas que não requerem muito conhecimento
 - Aritmética, repetição, ordenação...
- Humanos são bons em tarefas que requerem conhecimento intensivo
 - Reconhecer faces em figuras, diagnóstico médico, entendimento de linguagem natural...

Propósito da Inteligência Computacional

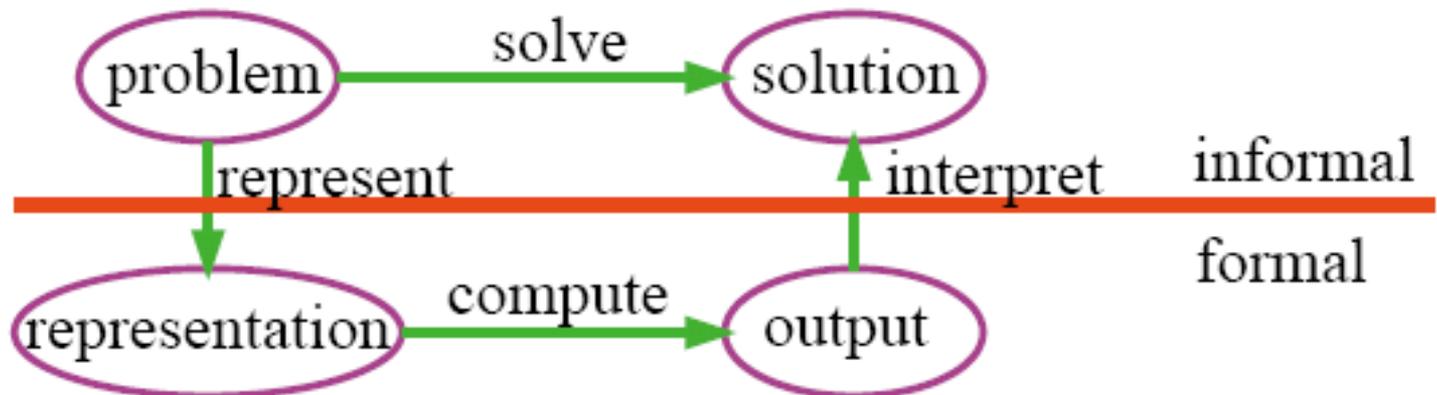


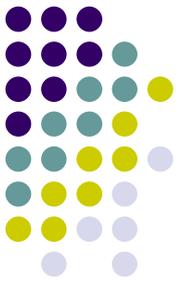
- Projetar sistemas computacionais que tem conhecimento sobre o mundo e agem nele baseados em conhecimento
 - Eles devem ser capazes de adquirir e representar conhecimento para resolver problemas
 - Adquirir e representar conhecimento sobre um domínio e usar este conhecimento para responder questões e resolver problemas

Sistemas de representação e raciocínio



- É composto de:
 - Uma linguagem para se comunicar com o computador
 - Uma forma de associar significado à linguagem
 - Procedimentos para computar respostas de acordo com a linguagem de entrada





Uma representação deve ser

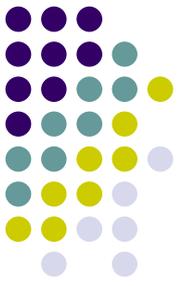
- Rica o bastante para expressar o conhecimento necessário para resolver o problema
- Ser tão próxima do problema quanto possível: compacta, natural e de fácil manutenção
- A computação sobre ela deve ser eficiente
- Permitir aprendizagem com os dados e com as experiências passadas
- Ter um balanceamento entre precisão e tempo de computação



Representação

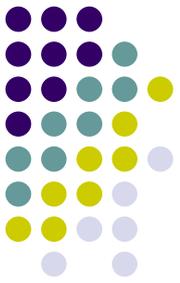
- Fortran, Java, C++, Lisp
 - Mapear um domínio para uma representação utilizando uma linguagem baseada em passos é uma tarefa difícil
- Linguagem Natural
 - O mapeamento se torna mais fácil, porém, a computação necessária para raciocinar sobre a entrada é muito difícil
- SRR
 - A distância entre a especificação natural e a representação do problema não estejam muito distantes
 - Dada alguma entrada, a computação necessária deve ser eficientemente determinada
- Exemplo simples de SRR: Banco de dados

Como usar um SRR



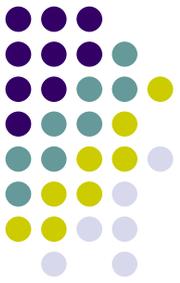
- Especifique o domínio de tarefa que você quer caracterizar
 - Qual é o **domínio** que você quer responder perguntas, resolver problema, ou interagir
- Distingua coisas que você quer falar sobre o domínio
 - Defina a **ontologia** do domínio
- Use símbolos para representar objetos e relações no domínio
 - Símbolos **denotam** objetos no mundo real.
- Coloque no computador o conhecimento sobre o domínio
 - O que é verdade no domínio (fatos – conhecimento declarativo), como resolver problemas no domínio (conhecimento procedural)
- Faça perguntas ao SRR

Componentes de um SRR



- A linguagem formal
 - Especifica as sentenças válidas que você pode usar para representar conhecimento
 - Define os símbolos válidos e como eles podem ser combinados
 - Uma linguagem é especificada como uma gramática
 - O conhecimento é um conjunto de sentenças nesta linguagem

Componentes de um SRR



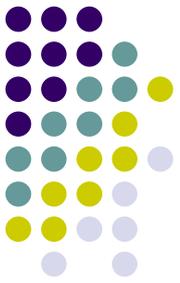
- A semântica
 - Especifica o significado das sentenças na linguagem
 - Especifica o comprometimento de como os símbolos na linguagem relatam o domínio de tarefa
 - Permite explicar o significado das sentenças em termos de domínio, assim como entender o significado das repostas
 - É importante que a semântica seja natural e intuitiva para que o resultado da computação seja facilmente interpretado

Componentes de um SRR



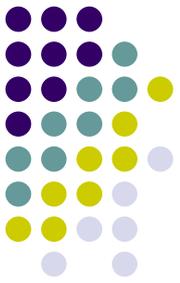
- A teoria de raciocínio ou teorema de prova
 - É uma especificação possivelmente não determinística de como uma resposta pode ser derivada da base de conhecimento
 - É tipicamente um conjunto de regras de inferência para encontrar consequências deixadas implícitas *ou*
 - É uma especificação não determinística de como uma resposta é computada
 - Procedimento de prova **íntegro** (*sound*) : somente gera respostas corretas com relação à semântica
 - Procedimento de prova **completo** (*complete*) : gera todas as respostas corretas ou garante produzir uma se ela existir

Implementação de um SRR



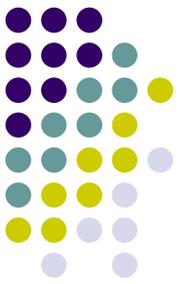
- Parser da linguagem
 - Distingue sentenças válidas da linguagem formal
 - Produz alguma forma interna de armazenamento como estruturas de dados no computador
- Procedimento de raciocínio
 - É a implementação da teoria de raciocínio juntamente com uma estratégia de busca
 - Resolver o não determinismo em uma teoria de raciocínio é papel da estratégia de busca
 - O procedimento de raciocínio manipula as estruturas de dados produzidas pelo parser, para produzir respostas

Implementação de um SRR

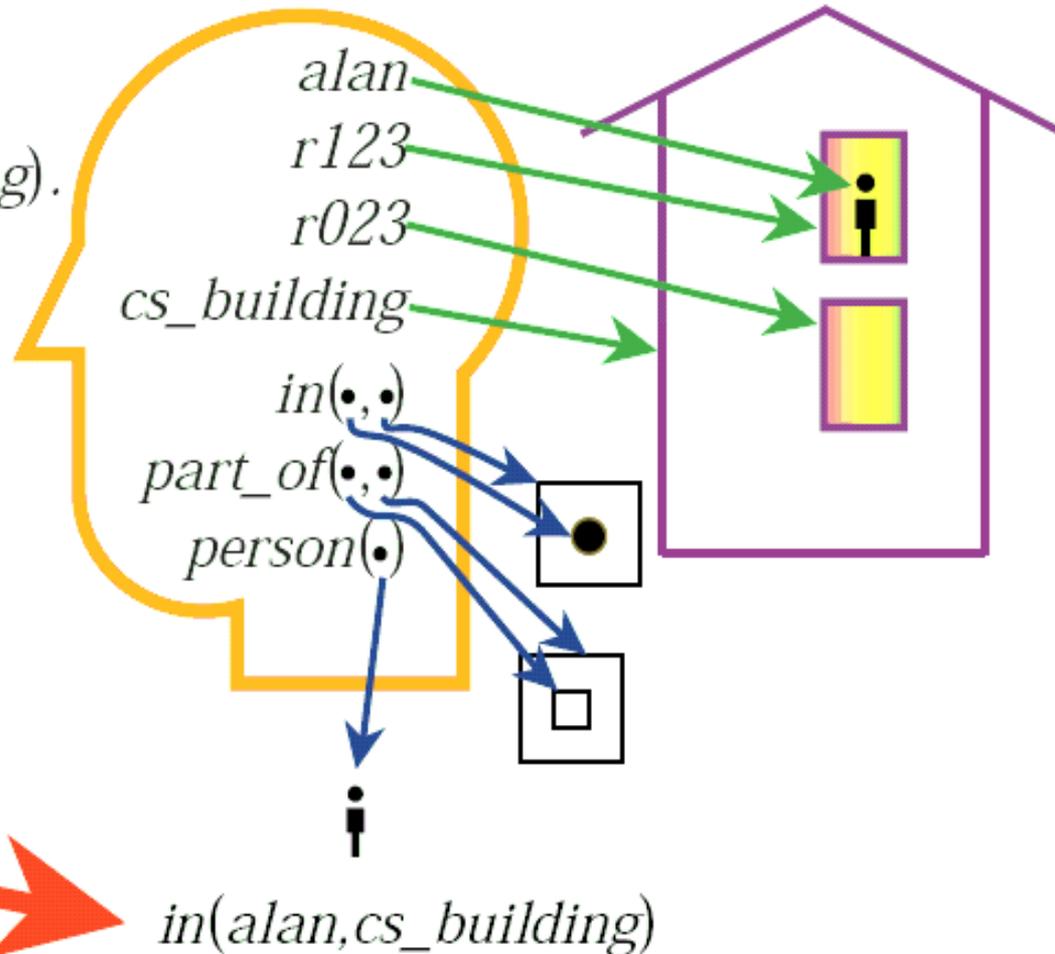
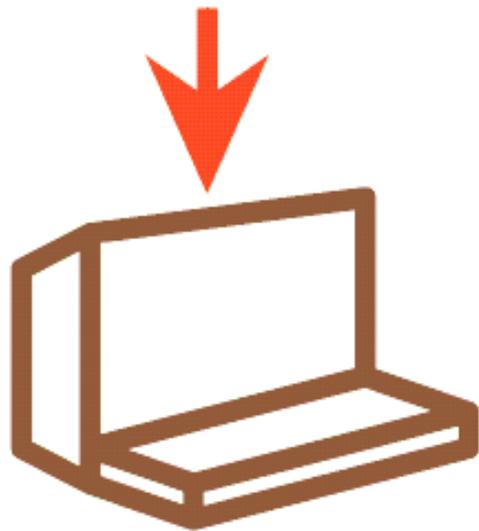


- Procedimento de raciocínio (continuação)
 - Podem existir muitas implementações diferentes para uma mesma teoria de raciocínio
 - Implementadas em diferentes linguagens, usando diferentes técnicas de codificação ou diferentes estratégias de busca
 - Pode ser incompleto
 - Porque existe alguma conclusão que semântica especifica que deveria existir, mas a teoria de raciocínio não pode derivar **ou**
 - Porque a estratégia de busca não pode encontrar uma resposta autorizada (*sanctioned*) pela teoria de raciocínio
 - A semântica não é refletida na implementação – ela dá significado aos símbolos por uma visão externa

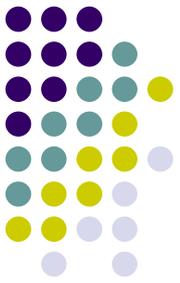
O significado dos símbolos está na cabeça do usuário



$in(alan, r123).$
 $part_of(r123, cs_building).$
 $in(X, Y) \leftarrow$
 $part_of(Z, Y) \wedge$
 $in(X, Z).$

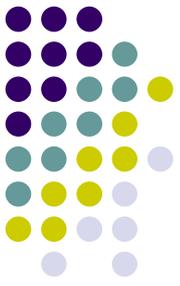


A lógica matemática



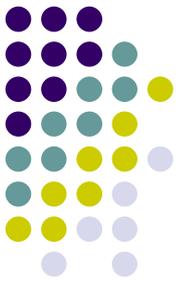
- Definição de semântica e teoria de raciocínio correspondem à noção de semântica Tarskiana e prova em lógica matemática
- A lógica permite
 - Definir conhecimento independente de como ele é usado
 - Verificar a corretude do conhecimento desde que conheçamos seu significado
 - Debater a verdade das sentenças da linguagem, dentro de um domínio e interpretação particular
- A lógica fornece a especificação do significado em termos de relacionamentos entre os símbolos no computador e o domínio de tarefa

Suposições para um SRR inicial



- Suposição sobre **Indivíduos e Relações**: O conhecimento do agente pode ser descrito em termos de indivíduos e as relações entre eles
 - Relações: nulas, unárias (propriedades) , binárias, ternárias...
- Suposição sobre o **Conhecimento Definido**: A base de conhecimento de um agente definições e declarações positivas
 - Declarações do tipo: “João está no quarto 123” ou “João está no laboratório”
 - Mas não declarações do tipo: “João não está no quarto 125”
 - O agente não conhece tudo

Suposições para um SRR inicial



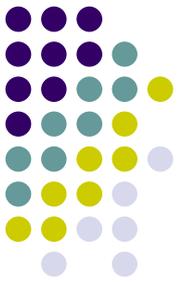
- Suposições sobre o **Ambiente Estático**: Nenhuma ação que muda o ambiente ocorre enquanto agente está construindo sua representação ou raciocinando
 - Permite ignorar o tempo
 - Sentenças do tipo: “João estava no quarto 123” são excluídas
- Suposições sobre o **Domínio Finito**: Existe um número finito de indivíduos de interesse no domínio e a cada indivíduo pode ser dado um nome único
- O SRR simples definido chama-se Datalog e utiliza cláusulas definidas (Prolog)

Sintaxe do Datalog



- **Variáveis** começam com letra maiúscula
 - X, Quarto, D1, The_big_cat...
- **Constantes** são palavras que começam com letras minúsculas ou uma seqüência de dígitos (numeral)
 - alan, r123, 125
- **Símbolos de predicados** são palavras que começam com letras minúsculas
 - parent, brother, colorof
- **Termos** são ou variáveis ou constantes
 - X, alan, D, Brother
- **Símbolos atômicos** (átomos) são da forma p ou $p(t_1, t_2, \dots, t_n)$
 - Onde p é um predicado e t_i são termos
 - ensina(maria, dc1221), feliz(X), pai(Bill, Y)

Sintaxe do Datalog

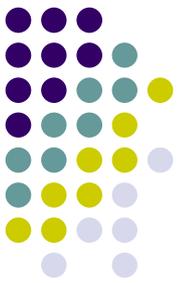


- **Cláusula definidas** são ou um símbolo atômico (fato) ou da forma:

$$\underbrace{a}_{\text{cabeça}} \leftarrow \underbrace{b_1 \wedge \dots \wedge b_m}_{\text{corpo}}$$

- onde a e b_i são símbolos atômicos
- Uma **query** é da forma $?b_1 \wedge \dots \wedge b_m$
- Uma **expressão** é ou um termos, ou um átomo, ou um cláusula definida, ou uma query
- A **base de conhecimento** é um conjunto de cláusulas definidas
- Comentários são precedidos de %

Exemplo de base de conhecimento



$in(alan, R) \leftarrow$
 $teaches(alan, cs322) \wedge$
 $in(cs322, R).$

$grandfather(william, X) \leftarrow$
 $father(william, Y) \wedge$
 $parent(Y, X).$