

## *Geração de Planos (Planning)*

May 14, 2015

# AI Planning

- Uma vez tendo estabelecido um conjunto de estados e construído um modelo completo e correto, precisamos traçar um *plano de ação* de forma que o agente possa seguir as ações e atingir os objetivos.
- Algoritmo básico:

```

function SIMPLE-PLANNING-AGENT(percept) returns an action
  static: KB, a knowledge base (includes action descriptions)
           p, a plan, initially NoPlan
           t, a counter, initially 0, indicating time
  local variables: G, a goal
                    current, a current state description

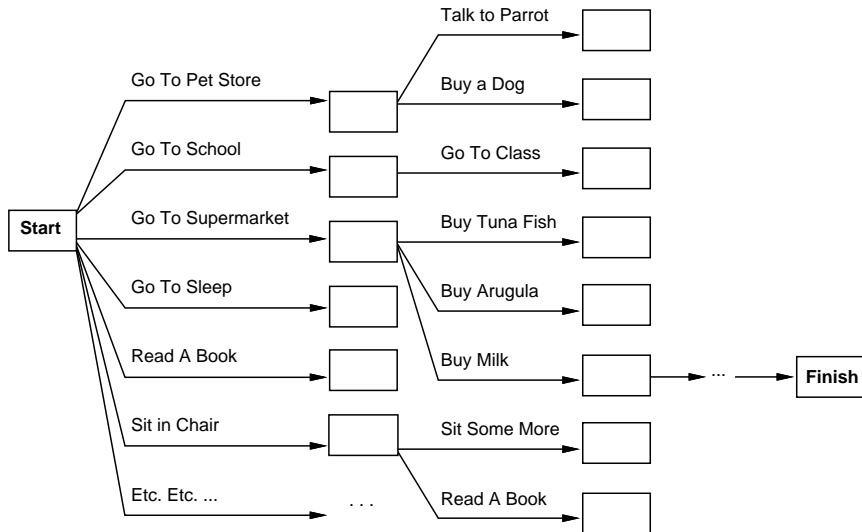
  TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
  current ← STATE-DESCRIPTION(KB, t)
  if p = NoPlan then
    G ← ASK(KB, MAKE-GOAL-QUERY(t))
    p ← IDEAL-PLANNER(current, G, KB)
  if p = NoPlan or p is empty then action ← NoOp
  else
    action ← FIRST(p)
    p ← REST(p)
  TELL(KB, MAKE-ACTION-SENTENCE(action, t))
  t ← t + 1
  return action

```

# AI Planning

- diferenças: representações de ações, estados, objetivos e planos.
- exemplo simples: comprar 1 l de leite, uma dúzia de bananas e um barbequim.
- problem solving:
  - ▶ estado inicial: agente está em casa, mas sem todas as coisas necessárias.
  - ▶ operadores: tudo que o agente pode fazer para obter os objetos.
  - ▶ opcional: heurística (ex, número de coisas ainda não compradas).

# AI Planning



# AI Planning

- três idéias principais: transparência de ações e estados, liberdade para adicionar ações ao plano de ações a qualquer momento (sem ser incremental), e independência das partes.
- Planning utilizando cálculo de situações:
  - ▶ **estado inicial:**  $At(Home, S_0) \wedge \neg Have(Milk, S_0) \wedge \neg Have(Bananas, S_0) \wedge \neg Have(Drill)$
  - ▶ **estado final:**  $\exists s At(Home, s) \wedge Have(Milk, s) \wedge Have(Bananas, s) \wedge Have(Drill, s)$
  - ▶ **operadores:** ex, axioma do estado sucessor para comprar leite:  $\forall a, s Have(Milk, Result(a, s)) \Leftrightarrow [(a = Buy(Milk) \wedge At(Supermarket, s)) \vee (Have(Milk, s) \wedge a \neq Drop(Milk))]$

# AI Planning

- Em “planning” representamos *sequências de ações*, não somente ações únicas. Em cálculo de situações representamos Result:
  - ▶  $\forall s \text{ Result}'([], s) = s$
  - ▶  $\forall a, p, s \text{ Result}'([a | p], s) = \text{Result}'(p, \text{Result}(a, s))$
- Uma solução para o problema das compras é um **plano**  $p$  que qdo aplicado ao estado inicial  $S_0$  produz a solução:
  - ▶  $\text{At}(\text{Home}, \text{Result}'(p, S_0)) \wedge \text{Have}(\text{Milk}, \text{Result}'(p, S_0)) \wedge$   
 $\text{Have}(\text{Bananas}, \text{Result}'(p, S_0)) \wedge \text{Have}(\text{Drill}, \text{Result}'(p, S_0))$
- ASK daria uma solução do tipo:
  - $p = [\text{Go}(\text{Supermarket}), \text{Buy}(\text{Milk}), \text{Buy}(\text{Bananas}),$   
 $\text{Go}(\text{HardwareStore}), \text{Buy}(\text{Drill}), \text{Go}(\text{Home})]$

# AI Planning

- soluções teóricas nem sempre são *eficientes* se usarmos regras de inferência não guiadas.
- para soluções práticas:
  - ▶ Restringir a linguagem de definição do problema.
  - ▶ Utilizar algoritmos de “planning” particulares (*planner* para cada problema, invés de algoritmos gerais para provas de teoremas).

# AI Planning

- Maioria dos “planners” utilizados descreve estados e operadores em STRIPS (linguagem baseada em cálculo de situações) ou extensões.
- Estados: representados por conjunções de literais “ground” (completamente instanciados), sem funções (predicados aplicados a símbolos constantes). Ex:  $At(Home) \wedge \neg Have(Milk) \wedge \neg Have(Bananas) \wedge \neg Have(Drill) \wedge \dots$
- Descrição do estado não precisa ser completa.
- Maior parte dos “planners” assume que se nada é dito sobre um literal, ele é falso (negação por falha em programação lógica).



# AI Planning

- Objetivos: tb representados por conjunções de literais. Ex:  $At(Home) \wedge Have(Milk) \wedge Have(Bananas) \wedge Have(Drill)$ .
- Podem conter variáveis. Ex, estar numa loja que venda leite:  $At(x) \wedge Sells(x, Milk)$ .
- Como em provadores automáticos de teoremas, assume-se que todas as variáveis do objetivo estão quantificadas existencialmente.
- Diferenças entre objetivos num “planner” e num provador de teoremas: objetivo no “planner” pergunta por uma sequência de ações que permitam validar o objetivo se este for executado. Objetivo no provador de teoremas é diretamente provado verdadeiro.

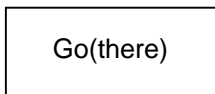
# AI Planning

- Representação de ações: operadores STRIPS consistem de três componentes:
  - ▶ **Descrição da ação:** para o “planner” serve apenas como um nome para a ação.
  - ▶ **Precondição:** conjunção de átomos (literais positivos) que devem ser verdadeiros antes do operador ser aplicado.
  - ▶ **Efeito (ou poscondição):** conjunção de literais (positivos ou negativos) que descrevem como a situação é modificada quando o operador é aplicado.

# AI Planning

- Ex:
  - Op( *ACAO* : *Go(there)*,
  - PRECON* : *At(there)  $\wedge$  Path(there, there)*,
  - EFEITO* : *At(there)  $\wedge$   $\neg$ At(there)*

*At(there), Path(there, there)*



*At(there),  $\neg$ At(there)*

# AI Planning

- Um operador pode conter variáveis: *esquema de operador*.
- Corresponde a uma família de ações.
- Somente operadores completamente instanciados podem ser executados.
- Linguagem de descrição de precondições e efeitos bastante restrita. Pode ser menos restrita.

# AI Planning

- Operador é dito *aplicável* num estado  $s$ , se há alguma forma de instanciar variáveis do operador tal que a precondição seja verdadeira para esta instância. No estado resultante, todos os literais positivos da poscondição também devem ser verdadeiros, exceto os negados.
- Ex: situação inicial,  
 $At(Home), Path(Home, Supermarket) \dots$ , ação  
 $Go(Supermarket)$  é aplicável, e a situação resultante contém  $\neg At(Home), At(Supermarket),$   
 $Path(Home, Supermarket) \dots$

# AI Planning

- Dada uma definição do problema em STRIPS, algoritmos de busca podem ser utilizados para resolver o problema de “planning” do estado inicial para o objetivo.
- Chamados geradores de planos sobre o *espaço de situações*, e *progressivos*.
- Problema com esta abordagem: fator de ramificação!
- Alternativa: gerador de planos *regressivo*.
- Algoritmo original utilizado por STRIPS era de um gerador de planos sobre o espaço de situações regressivo, mas era incompleto, pois não sabia lidar com o fato de gerar uma conjunção.
- Ineficiente produzir algoritmo completo utilizando esta abordagem de geração de planos por situações.

# AI Planning

- Alternativa: busca pelo *espaço de planos*.
- Começamos com um plano simples, incompleto: *plano parcial*.
- Aplicamos os operadores (de plano) e geramos outros planos até chegar a um que nos leve à solução.
- Possíveis operadores de plano: adicionar um novo passo ao plano, imposição de ordem aos passos do plano, instanciação de uma variável etc.
- Solução é o plano final. Caminho para chegar a este plano é irrelevante (eficiência...)

# AI Planning

- Operações sobre planos:
  - ▶ **Operadores de refinamento:** adicionam restrições a um plano parcial. Eliminam alguns planos do conjunto possível de planos.
  - ▶ **Operadores de modificação:** qualquer outro operador que não seja de refinamento.
- Ex: Plano para calçar um par de sapatos.
- Objetivo:  
*ColocadoSapatoPeDireito  $\wedge$  ColocadoSapatoPeEsquerdo.*
- Estado inicial: não necessário.



# AI Planning

- Operadores:
  - ▶ *Op(ACAO : Colocar Sapato Pe Direito, PRECOND : Colocada Meia Pe Direito, EFEITO : Colocado Sapato Pe Direito)*
  - ▶ *Op(ACAO : Colocar Meia Pe Direito, EFEITO : Colocada Meia Pe Direito)*
  - ▶ *Op(ACAO : Colocar Sapato Pe Esquerdo, PRECOND : Colocada Meia Pe Esquerdo, EFEITO : Colocado Sapato Pe Esquerdo)*
  - ▶ *Op(ACAO : Colocar Meia Pe Esquerdo, EFEITO : Colocada Meia Pe Esquerdo)*

# AI Planning

- Plano parcial deve ter dois passos: Colocar Sapato Pe Direito, Colocar Sapato Pe Esquerdo. Quem vem primeiro?
- *Menos “comprometedor”*: somente escolher planos com informações relevantes no momento. Deixar outras escolhas para serem feitas mais tarde.
- *Plano de ordem parcial*: alguns passos do plano são ordenados, outros não.
- *Plano de ordem total*: todos os passos são ordenados.
- *Linearização* de planos: geração de um plano de ordem total após adição de restrições ao plano anterior.
- Restrições no valor das variáveis a serem instanciadas.

# AI Planning

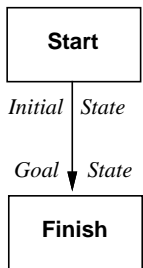
- Um plano é formalmente definido como uma estrutura de dados contendo os seguintes 4 componentes:
  - ▶ **conjunto de passos.** Cada passo é um dos operadores do problema.
  - ▶ **conjunto de restrições de ordenação** dos passos.
  - ▶ **conjunto de instâncias de variáveis** para cada passo.
  - ▶ **conjunto de ligações causais.** Ex:  $S_i \rightarrow^c S_j$ , com  $c$  sendo precondição de  $S_j$ .
- Plano inicial descreve o problema ainda não resolvido. Consiste de dois passos: início e fim com relação de precedência Início  $\prec$  Fim.

# AI Planning

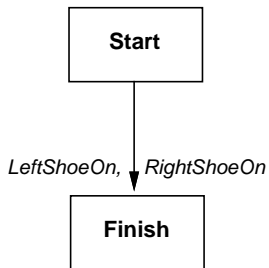
- Passo Início não tem precondições e o efeito é adicionar todas as proposições verdadeiras ao estado inicial.
- Passo Fim tem o objetivo como precondição, e nenhum efeito.
- Ex:

```
Plan(STEPS: { S1: Op(ACAO:Inicio),
              S2: Op(ACAO: Fim,
                    PRECOND: ColocadoSapatoPeDireito ∧
                              ColocadoSapatoPeEsquerdo) },
      ORDERINGS: { S1 < S2 },
      BINDINGS: {},
      LINKS: {})
```

# AI Planning



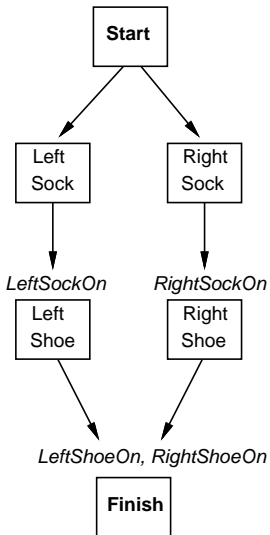
(a)



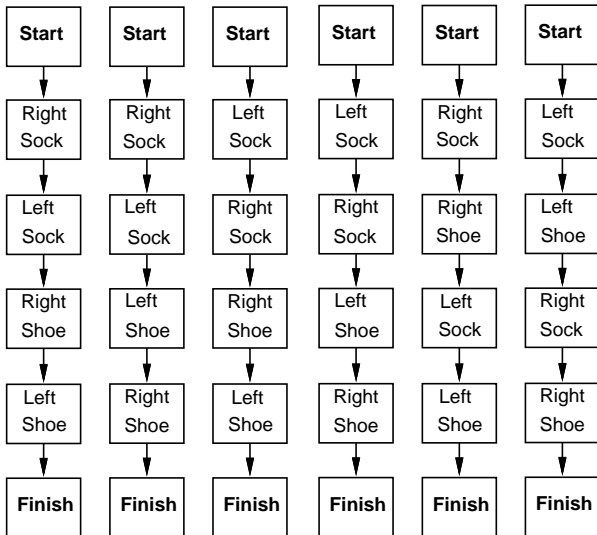
(b)

# AI Planning

**Partial Order Plan:**



**Total Order Plans:**



# AI Planning

Soluções:

- uma solução é um plano que um agente possa executar, e que garanta chegar ao objetivo.
- para garantir que um plano é uma solução, basta garantir que o plano é totalmente ordenado e instanciado.
- na prática, não razoável usar esta abordagem.
  - ▶ 1) mais natural gerar um plano de ordem parcial (mais geral).
  - ▶ 2) alguns agentes podem executar em paralelo.
  - ▶ 3) mais flexível para combinar com outros planos.

# AI Planning

- Solução deve ser **consistente** e **completa**.
- Um plano é completo se toda pré-condição de cada passo for satisfeita por algum outro passo.
- mais formalmente: um passo  $S_i$  atinge uma pré-condição  $c$  do passo  $S_j$  se:
  - ▶ 1)  $S_i \prec S_j$  e  $c \in EFEITOS(S_i)$
  - ▶ 2) não há nenhum passo  $S_k$  tal que  $(\neg c) \in EFEITOS(S_k)$ , onde  $S_i \prec S_k \prec S_j$  em alguma linearização do plano.



# AI Planning

- Um plano é consistente se não tiver contradições na ordem ou valores das restrições.
- Contradição ocorre quando  $S_i \prec S_j$  aparece ao mesmo tempo que  $S_j \prec S_i$  no plano gerado ou quando  $v = A$  e  $v = B$ .
- Como  $\prec$  e  $=$  são operadores relacionais com propriedade transitiva, teremos contradição se  $S_1 \prec S_2$ ,  $S_2 \prec S_3$ , e  $S_3 \prec S_1$ .

# AI Planning

## Exemplo de Planning com Ordem Parcial

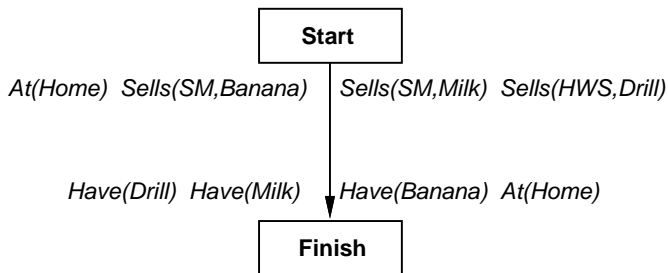
- Planner regressivo usando ordenação parcial, buscando soluções no espaço de planos.
- Começa com um plano inicial representado pelos passos inicial e final.
- Adiciona passos a cada iteração.
- Se o passo for inconsistente, retrocede (*backtracks*) e tenta outro ramo no espaço de busca.
- Para manter a busca guiada, o planner somente considera adicionar passos cuja pré-condição ainda não tenha sido alcançada (links causais são utilizados neste caso).

# AI Planning

## Exemplo de Planning com Ordem Parcial

- Op(ACAO: Inicia,  
 EFEITO:  $\text{At}(\text{Home}) \wedge \text{Sells}(\text{HWS}, \text{Drill}) \wedge$   
 $\text{Sells}(\text{SM}, \text{Milk}) \wedge \text{Sells}(\text{SM}, \text{Bananas}))$
- Op(ACAO: Termina,  
 PRECOND:  $\text{Have}(\text{Drill}) \wedge \text{Have}(\text{Milk}) \wedge$   
 $\text{Have}(\text{Bananas}) \wedge \text{At}(\text{Home}))$
- Op(ACAO: Go(there),  
 PRECOND:  $\text{At}(\text{here})$   
 EFEITO:  $\text{At}(\text{there}) \wedge \neg \text{At}(\text{here}))$
- Op(ACAO: Buy(x),  
 PRECOND:  $\text{At}(\text{shop}) \wedge \text{Sells}(\text{shop}, x)$   
 EFEITO:  $\text{Have}(x)$

# AI Planning

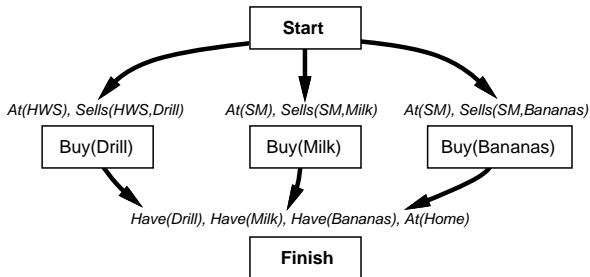
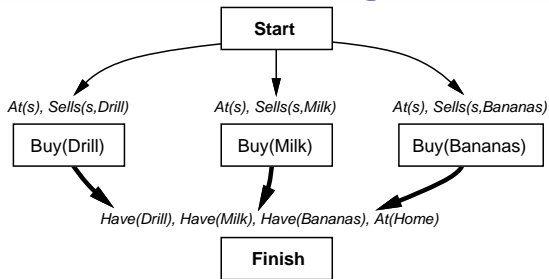


# AI Planning

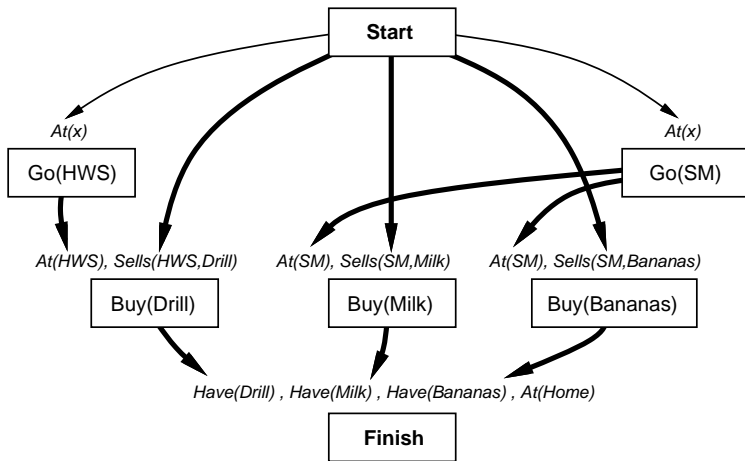
## Exemplo de Planning com Ordem Parcial

- Várias maneiras de gerar planos para este problema.
- Algumas levam a caminhos que não são soluções.
- Planner escolhe comprar somente aquelas coisas que interessam, dos lugares que interessam, sem gastar muito tempo.
- **Links protegidos**, um link causal é protegido garantindo que 'ameaças', isto é, passos que poderiam 'deletar' uma condição protegida, são ordenados de forma a aparecer antes ou depois do link protegido: **demotion** ou **promotion**.

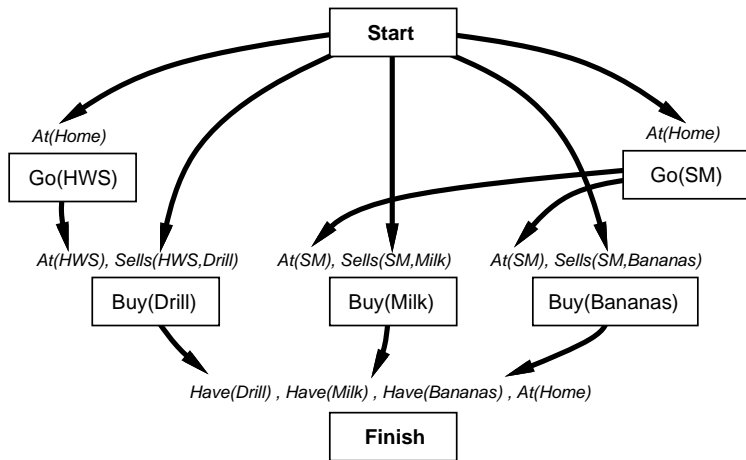
# AI Planning



# AI Planning



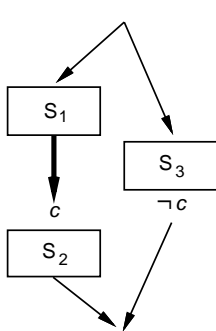
# AI Planning



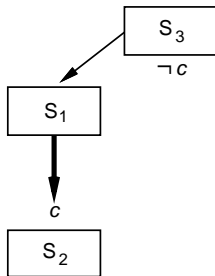


# AI Planning

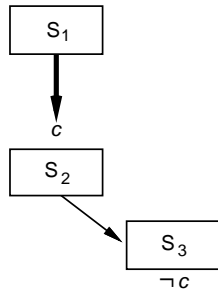
## Resolução de Conflitos



(a)



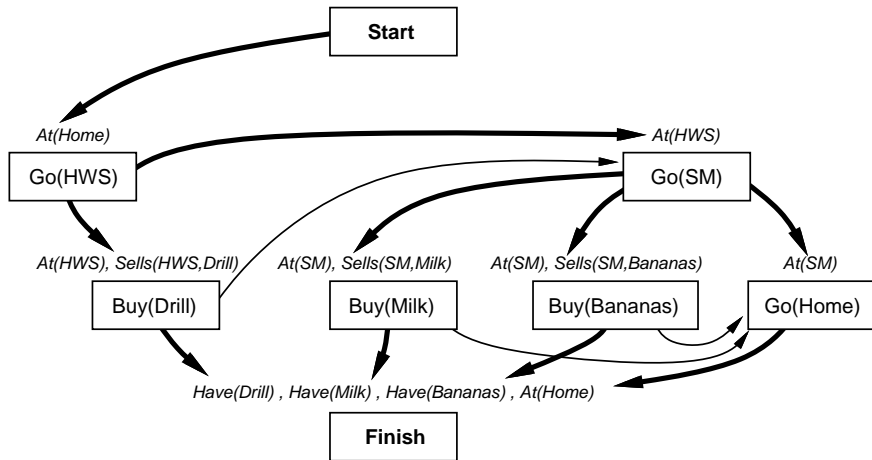
(b)



(c)

# AI Planning

## Exemplo de Planning com Ordem Parcial



# AI Planning

