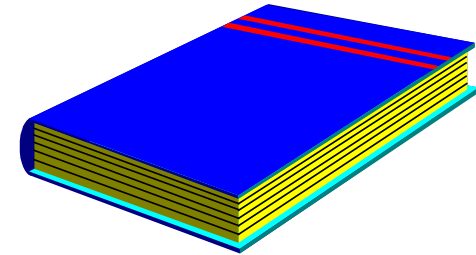


Planeamento



- **Bibliografia:**

- **Artificial Intelligence: a Modern Approach**
Stuart RUSSEL e Peter NORVIG
Prentice Hall, 1994
- **An Introduction to Least Commitment Planning**
Daniel S. WELD
AI Magazine, Winter, 1994

Planeamento

- **Dados**

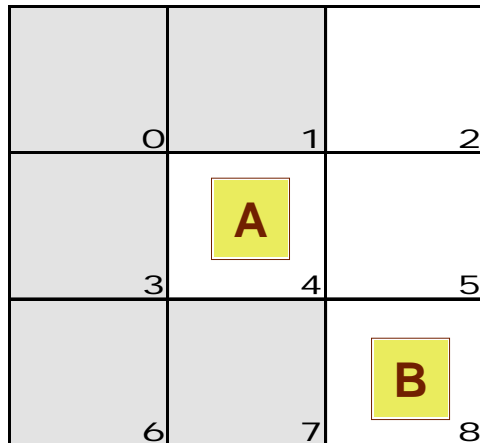
- um Estado Inicial
- um conjunto de Acções
- um Estado Objectivo
-

- **Determinar**

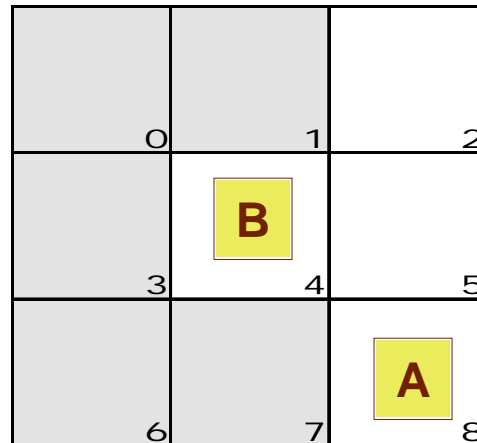
- sequência de Acções que permite atingir o objectivo

Problema-Tipo 1

● Planeamento de trajectórias



Estado Inicial



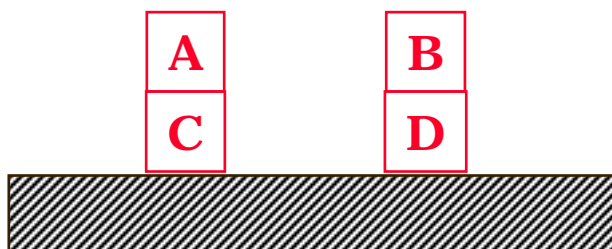
Estado-objectivo

Operações:

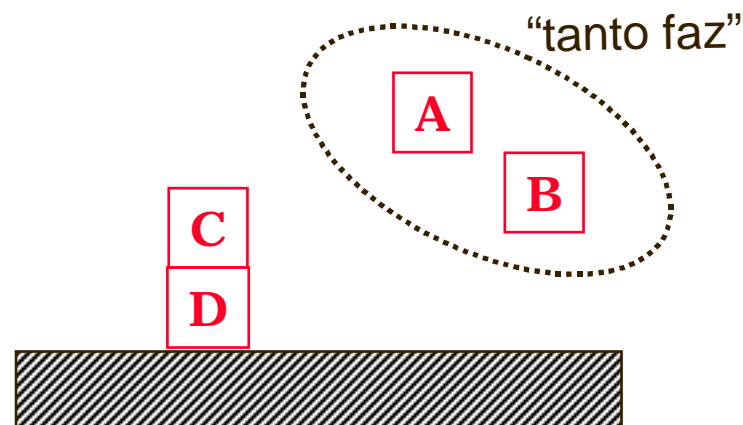
- move block A up
- move block A down
- move block A right
- move block A left
- move block B up
- move block B down
- move block B right
- move block B left

Problema-Tipo 2

- **Mundo de blocos**

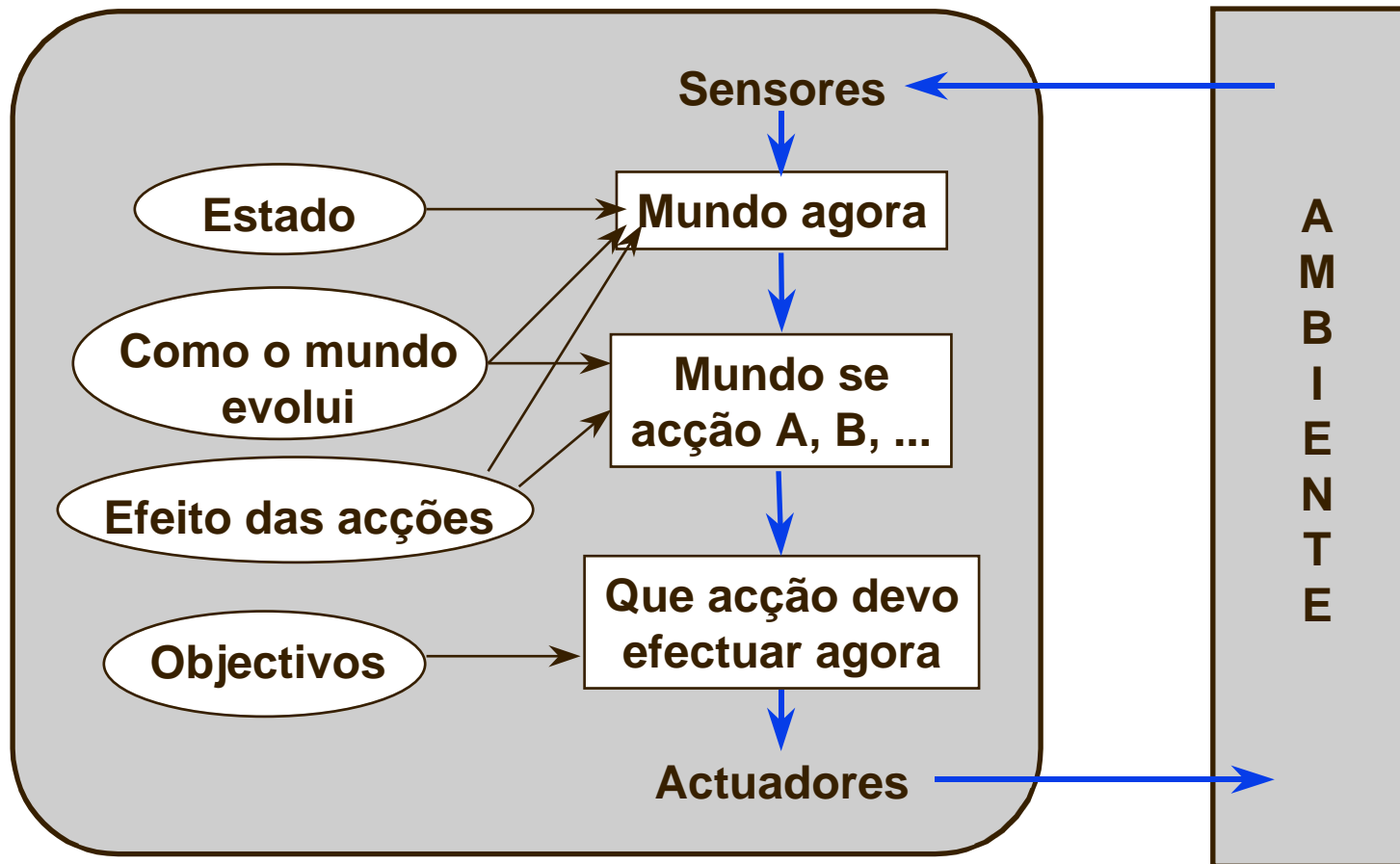


Estado Inicial



Estado-objectivo

Agente guiado por objectivos



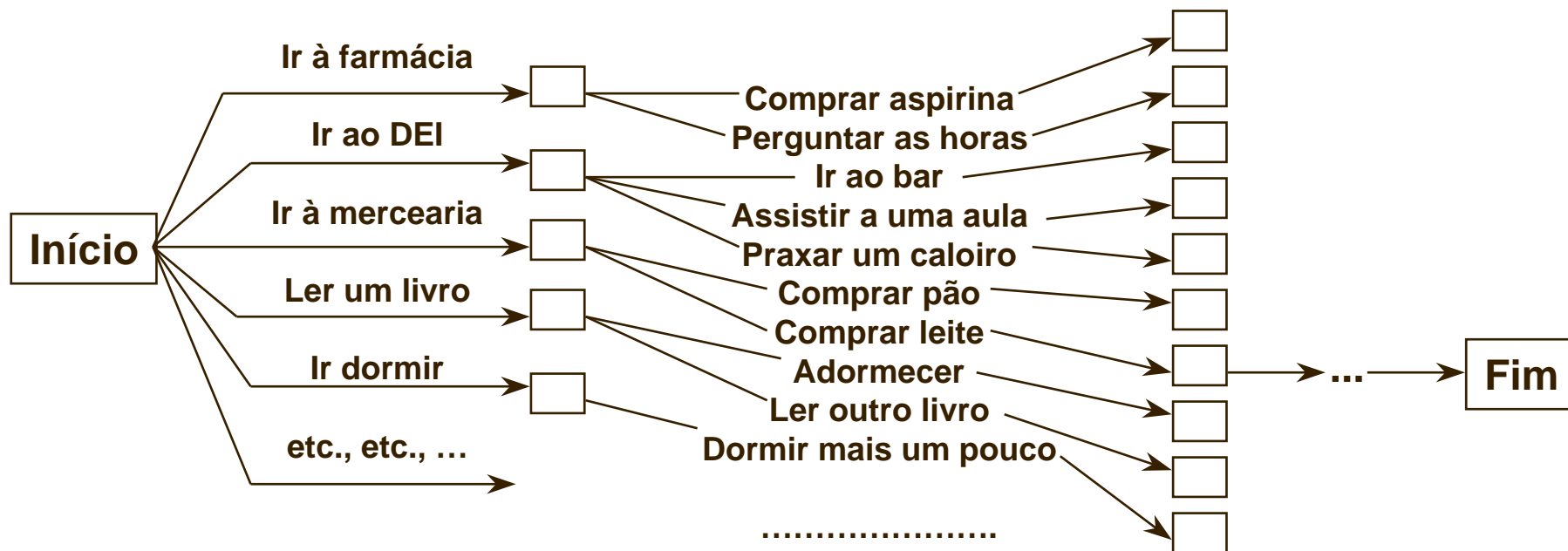
Planeamento como Procura num Espaço de Estados

- **Representação num agente que resolve problemas procurando num espaço de estados:**
 - **Acções:** programas que geram descrições de estados-sucessores
 - **Estados:** descrições completas
 - **Objectivos:** teste de objectivo (“caixa negra”) + função heurística
 - **Planos:** sequências não interrompidas de acções

Planeamento como Procura num Espaço de Estados

● Planear:

“Ter um pacote de leite, um cacho de bananas e um berbequim”



Planeamento como Procura num Espaço de Estados

- **Planeamento requer mais flexibilidade:**
 - “pegar” numa parte do problema
 - representações parciais dos estados
 - inserção de acções em qualquer ponto do plano
- **Felizmente...**
 - muitas parcelas do mundo são bastante independentes umas das outras

Dividir para reinar

- ☞ São necessários algoritmos específicos, dependentes do tipo de problema
- ☞ Sub-problemas são quase sempre mais simples de resolver

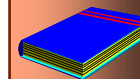
- ☐ Sub-objectivos independentes:
Planeamento linear

- ☐ Sub-objectivos pouco independentes:
Planeamento não-linear

Representação da Mudança no Mundo

- **Manter um modelo interno do mundo**
 - Frases sobre o estado actual do mundo
 - Frases actualizadas em face de novas percepções e de acções realizadas
- **Regras diacrónicas:**
 - regras que descrevem a forma como o mundo muda

Como representar diferentes acções e situações numa mesma base de conhecimento?



Cálculo Situacional

- Lógica de 1ª ordem
- O mundo visto como uma sequência de **situações**

Cálculo Situacional

- Lógica de 1ª ordem
- O mundo visto como uma sequência de **situações**

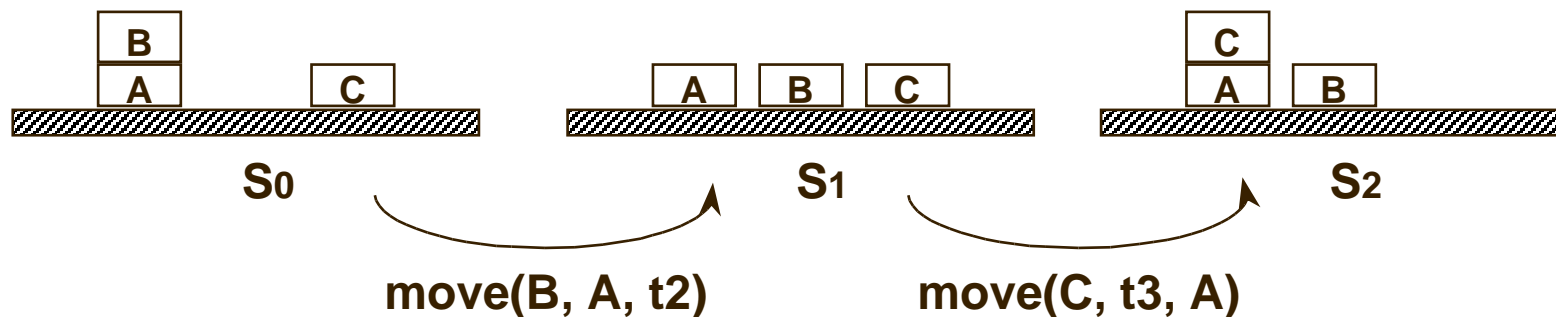
“instantâneos” do estado do mundo

Cálculo Situacional

- Lógica de 1ª ordem
- O mundo visto como uma sequência de **situações**

“instantâneos” do estado do mundo

- **Nova situação:** gerada a partir de situação anterior, pela aplicação de uma acção



Cálculo Situacional

- Representação:

- Relações e propriedades que mudam com o tempo ganham um argumento:

on(B, A, S₀) on(A, t1, S₀) on(C, t3, S₀)
clear(B, S₀) clear(C, S₀) clear(t2, S₀)

← Situação S₀

Cálculo Situacional

- **Representação:**

- **Relações e propriedades que mudam com o tempo ganham um argumento:**

on(B, A, S₀) on(A, t₁, S₀) on(C, t₃, S₀)
clear(B, S₀) clear(C, S₀) clear(t₂, S₀)

← Situação S₀

- **Mudança de uma situação para outra:**

- função “result(a, s)”

- result(move(B, A, t₂), S₀) = S₁

Cálculo Situacional

□ Acções descritas

○ por axiomas de efeito:

x, y, z, s $\text{clear}(x, s)$ $\text{on}(x, y, s)$ $\text{clear}(z, s) ?$
? $\text{clear}(y, \text{result}(\text{move}(x, y, z), s))$

x, y, z, s $\text{clear}(x, s)$ $\text{on}(x, z, s)$ $\text{clear}(y, s) ?$
? $\text{clear}(y, \text{result}(\text{move}(x, z, y), s))$

Cálculo Situacional

□ Acções descritas

- por axiomas de efeito:

x, y, z, s $\text{clear}(x, s)$ $\text{on}(x, y, s)$ $\text{clear}(z, s) ?$
 $? \text{clear}(y, \text{result}(\text{move}(x, y, z), s))$

x, y, z, s $\text{clear}(x, s)$ $\text{on}(x, z, s)$ $\text{clear}(y, s) ?$
 $? \text{clear}(y, \text{result}(\text{move}(x, z, y), s))$

Mas...

...não basta especificar o que muda, é necessário especificar o que não muda

Cálculo Situacional

□ Acções descritas também

- por axiomas de persistência:

$x, y, z, s \text{ clear}(y, s) \quad [(a ? \text{move}(x, z, y))$
 $(\text{clear}(x, s) \text{ on}(x, z, s))] ? \text{clear}(y, \text{result}(a, s))$

$x, y, z, s \text{ clear}(y, s) \quad [(a ? \text{move}(x, y, z))$
 $(\text{clear}(x, s) \text{ on}(x, y, s))] ? \text{clear}(y, \text{result}(a, s))$

Cálculo Situacional

□ Acções descritas também

- por axiomas de persistência:

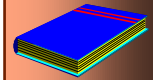
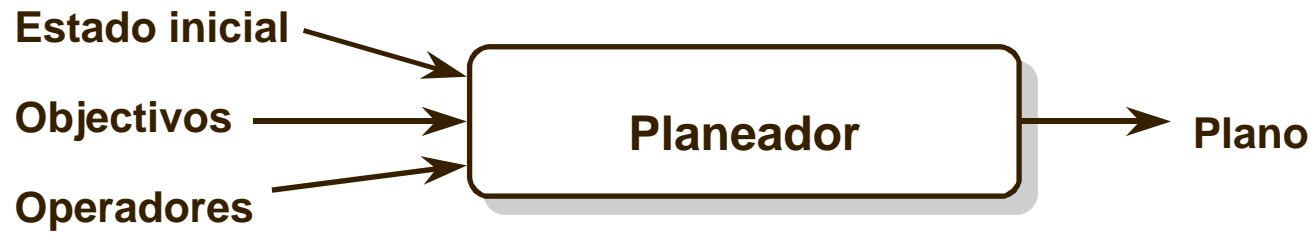
x, y, z, s $\text{clear}(y, s) \quad [(a \text{ ? move}(x, z, y))$
 $\quad (\text{clear}(x, s) \quad \text{on}(x, z, s))] \text{ ? } \text{clear}(y, \text{result}(a, s))$

x, y, z, s $\text{clear}(y, s) \quad [(a \text{ ? move}(x, y, z))$
 $\quad (\text{clear}(x, s) \quad \text{on}(x, y, s))] \text{ ? } \text{clear}(y, \text{result}(a, s))$

Problema:

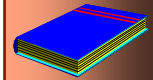
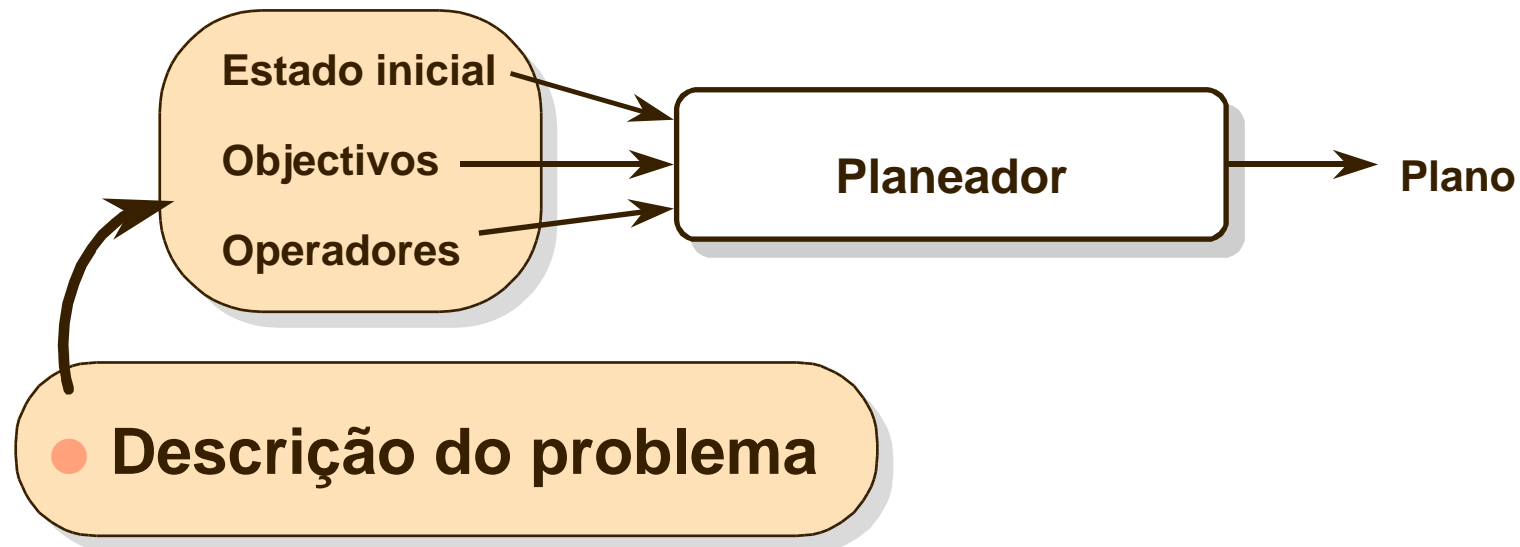
número elevado de axiomas de persistência necessários

Planeamento com Cálculo Situacional



RUSSEL e NORVIG, Cap. 11

Planeamento com Cálculo Situacional



RUSSEL e NORVIG, Cap. 11

Planeamento com Cálculo Situacional



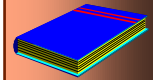
- **Descrição do problema:**

- **Estado inicial:**

- frase lógica arbitrária sobre uma situação S_0

- **Objectivos:** “query” lógica sobre situações

- **Operadores:** conjunto de descrições lógicas de acções



Cálculo Situacional em Planeamento

- **Planear:**

“Ter um pacote de leite, um cacho de bananas e um berbequim”

- **Descrição do problema:**

- Estado inicial**

- frase lógica arbitrária sobre uma situação S_0

em(casa, s_0)

ter(leite, s_0)

ter(bananas, s_0)

ter(berbequim, s_0)

Cálculo Situacional em Planeamento

- **Planear:**

“Ter um pacote de leite, um cacho de bananas e um berbequim”

- **Descrição do problema:**

- **Objectivos**

- “query” lógica sobre situações

s em(casa,s)

ter(leite,s) ter(bananas,s) ter(berbequims)

Cálculo Situacional em Planeamento

- **Planear:**

“Ter um pacote de leite, um cacho de bananas e um berbequim”

- **Descrição do problema:**

- **Operadores**

- **conjunto de descrições lógicas de acções**

a,s ter(leite,result(a,s))
[(a = comprar(leite)) em(mercearias,s)
(ter(leite,s) (a ? verter(leite)))]

Cálculo Situacional em Planeamento

- Dava jeito que o “result” pudesse lidar com seqüências de situações...

- **Result(l,s):**

$s \text{ result } ([],s) = s$

$a,p,s \text{ result } ([a|p],s) = \text{result } (p,\text{result}(a,s))$

- **Uma solução para o problema é um plano “p” tal que**

$\text{em}(\text{casa}, \text{result } (p,s_0)) \quad \text{ter}(\text{leite}, \text{result } (p,s_0))$
 $\text{ter}(\text{bananas}, \text{result } (p,s_0)) \quad \text{ter}(\text{berbequim}, \text{result } (p,s_0))$

Cálculo Situacional em Planeamento

- Uma solução para o problema é um plano “p” tal que

$\text{em}(\text{casa}, \text{result}(p, s_0)) \wedge \text{ter}(\text{leite}, \text{result}(p, s_0))$
 $\wedge \text{ter}(\text{bananas}, \text{result}(p, s_0)) \wedge \text{ter}(\text{berbequim}, \text{result}(p, s_0))$

- Possível solução:

$p = [\text{ir}(\text{mercearia}), \text{comprar}(\text{leite}), \text{comprar}(\text{bananas}),$
 $\text{ir}(\text{loja ferragens}), \text{comprar}(\text{berbequim}), \text{ir}(\text{casa})]$

Cálculo Situacional em Planeamento

- **Problema da persistência**

- **problema representacional:**

- número elevado de axiomas necessários



- axiomas de estado sucessor

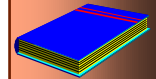
- **problema inferencial:**

- “transportar” um valor que não se altera ao longo de uma cadeia de inferência longa



- registar apenas o que se altera

- sistemas de raciocínio específicos



RUSSEL e NORVIG, Secção 7.6

Cálculo Situacional em Planeamento

- **Problema da qualificação:**

- quais as condições para que uma dada acção garantidamente se possa executar?

- **Problema da ramificação:**

- proliferação de consequências implícitas das acções



- ➔ axiomas específicos
- ➔ sistemas de raciocínio específicos

Cálculo Situacional em Planeamento

- **Outras limitações do Cálculo Situacional**
 - **Situações são pontos instantâneos no tempo**

Cálculo Situacional em Planeamento

● Limitações do Cálculo Situacional

- Situações são pontos instantâneos no tempo

- Vocacionado para mundos em que ocorre uma acção de cada vez:
 - Múltiplas acções em simultâneo ou múltiplos agentes no mundo, exigem que se definam acções compostas
 - Não pode ser usado quando as acções têm durações diferentes ou os seus efeitos dependem da duração

Cálculo Situacional em Planeamento

● Limitações do Cálculo Situacional

- Vocacionado para mundos em que ocorre uma acção de cada vez:
 - Múltiplas acções em simultâneo ou múltiplos agentes no mundo, exigem que se definam acções compostas
 - Não pode ser usado quando as acções têm durações diferentes ou os seus efeitos dependem da duração

- A utilização de um demonstrador de teoremas genérico torna-se ineficiente

Cálculo Situacional em Planeamento

- **Limitações do Cálculo Situacional**

- A utilização de um demonstrador de teoremas genérico torna-se ineficiente
- Podem ser introduzidos passos irrelevantes no plano

Cálculo Situacional em Planeamento

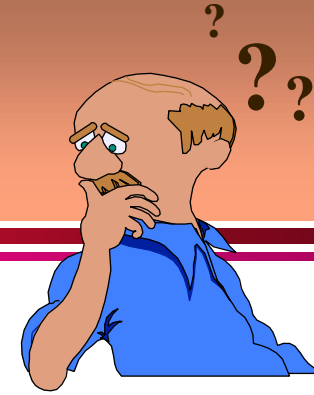
● Limitações do Cálculo Situacional

- ❑ A utilização de um demonstrador de teoremas genérico torna-se ineficiente
- ❑ Podem ser introduzidos passos irrelevantes no plano

... apesar de tudo, é suficiente para muitos domínios de planeamento...

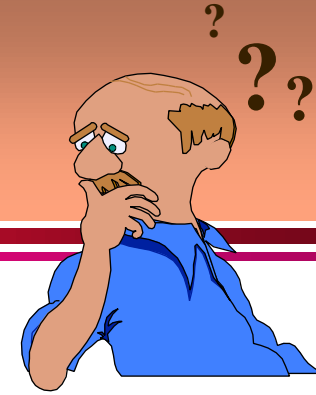
... mas não para os mais interessantes.

Que fazer?



- **Utilizar linguagem mais restritiva**
 - para diminuir o número de soluções onde procurar
- **Recorrer a algoritmos específicos**
 - algoritmos especialmente concebidos para processar as linguagens restritas de forma eficiente

Que fazer?



- **Exemplo de linguagem:**

STRIPS

- **Mantém muita da expressividade do cálculo situacional**
- **Existem para ela algoritmos de planeamento eficientes**

STRIPS

● Operadores

Descrevem acções

□ Pré-condições

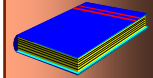
- Têm que verificar-se para o operador ser aplicável, logo para que a acção possa ser executada

□ Delete-list

- Predicados que se tornam falsos após execução da acção

□ Add-list

- Predicados que se tornam verdadeiros após execução da acção



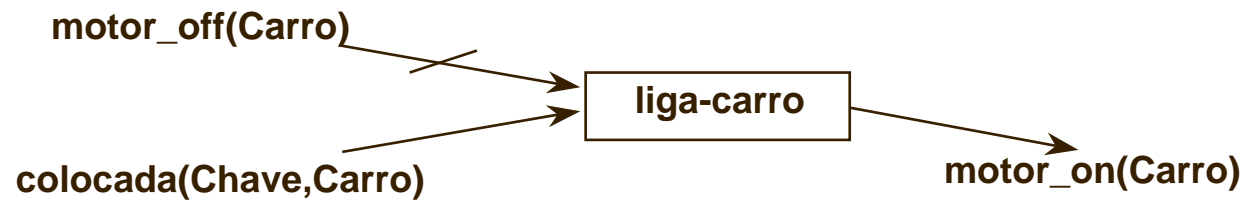
STRIPS

● Exemplo

□ Forma escrita

- Nome: liga-carro
- precond: motor_off(Carro) colocada(Chave, Carro)
- del-list: motor_off(Carro)
- add-list: motor_on(Carro)

□ Graficamente



STRIPS - Representação

- **Estados**

- **Conjunções de literais sem variáveis**

- **Objectivos**

- **Conjunções de literais**
- **Podem conter variáveis (assumem-se existencialmente quantificadas)**

STRIPS - Representação

- **Acções (operadores)**

- **Pré-condições, Delete-list, Add-list:**

- **Conjunções de literais positivos**

Os operadores podem conter variáveis:

- **Operator Schemas → representam famílias de acções**

- **As variáveis assumem-se universalmente quantificadas**
 - **Geralmente, exige-se que as variáveis estejam instanciadas no momento da execução**

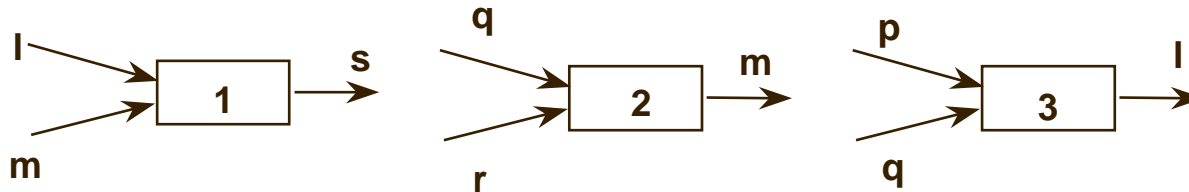
STRIPS e Persistência

- **Assumpção STRIPS**

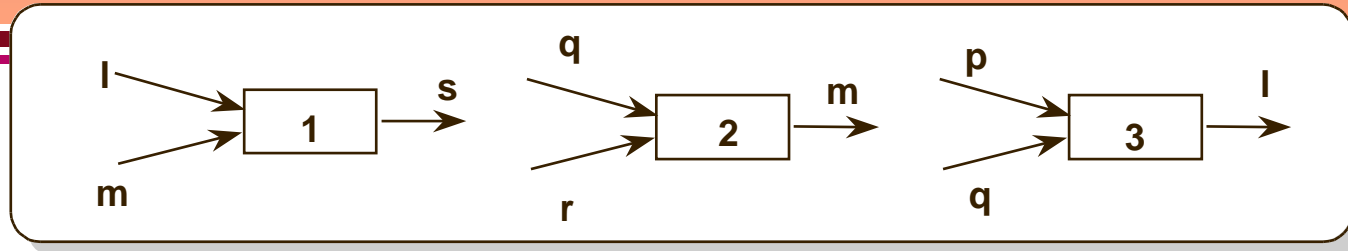
- **Tudo o que não for retirado explicitamente por um operador mantém-se válido no estado resultante da aplicação desse operador**

Como o STRIPS funciona

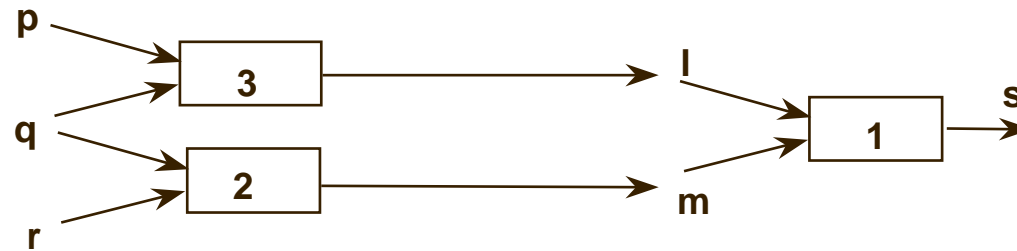
- Estado inicial: p q r
- Objectivo: s
- Operadores:



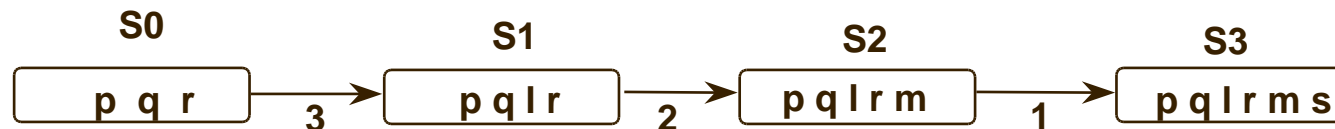
Como o STRIPS funciona



- Construir a árvore de procura (backward chaining)

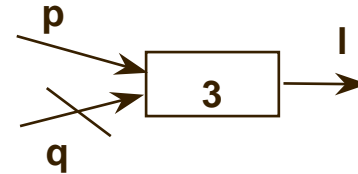


- Aplicar os operadores (forward chaining)

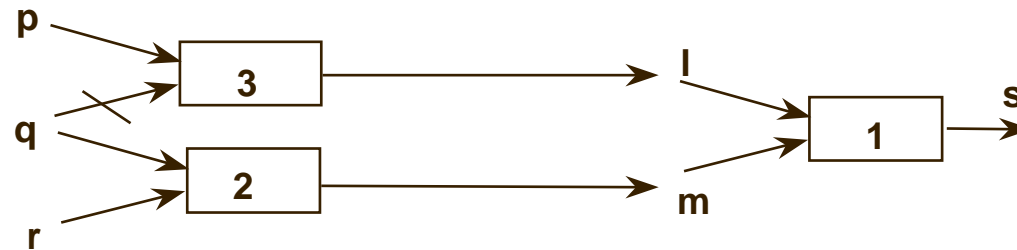


Como o STRIPS funciona

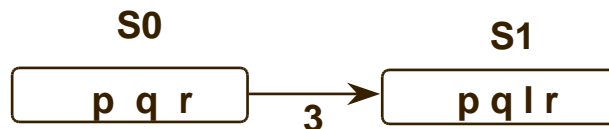
E se o operador 3 elimina q?



- Construir a árvore de procura (backward chaining)

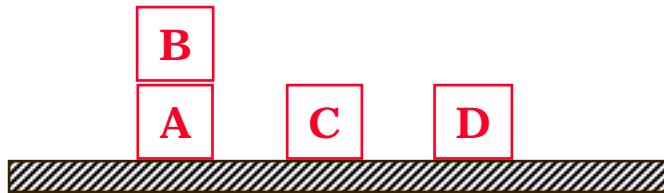


- Aplicar os operadores (forward chaining)



**INTERACÇÃO
ENTRE OBJECTIVOS !**

Exemplo



Estado Inicial

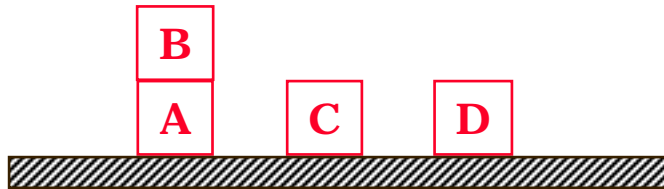
on(B,A)
clear(B)
clear(C)
clear(D)
ontable(A)
ontable(C)
ontable(D)
armempty



Estado Final

on(C,A)
on(B,D)
ontable(A)
ontable(D)

Exemplo



● Operadores:

STACK(x,y)

P: clear(y), holding(x)

D: clear(y), holding(x)

A: armempty, on(x,y)

PUTDOWN(x)

P: holding(x)

D: holding(x)

A: armempty, ontable(x)

UNSTACK(x,y)

P: on(x,y), clear(x), armempty

D: on(x,y), armempty

A: holding(x), clear(y)

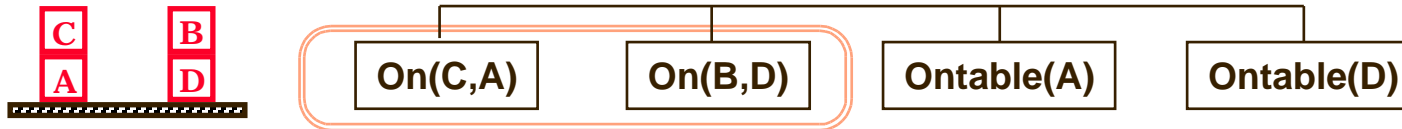
PICKUP(x)

P: ontable(x), clear(x), armempty

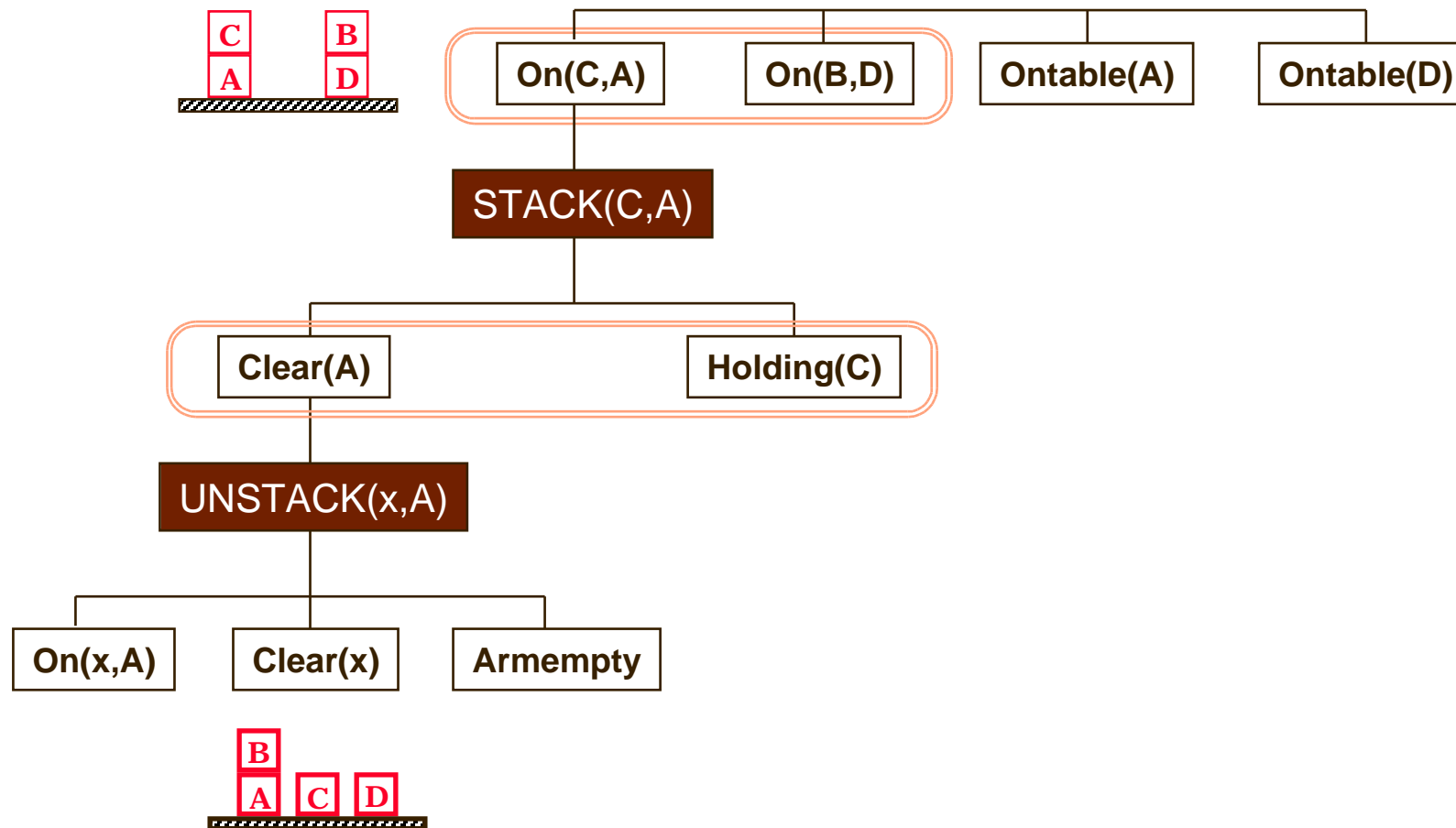
D: ontable(x), armempty

A: holding(x)

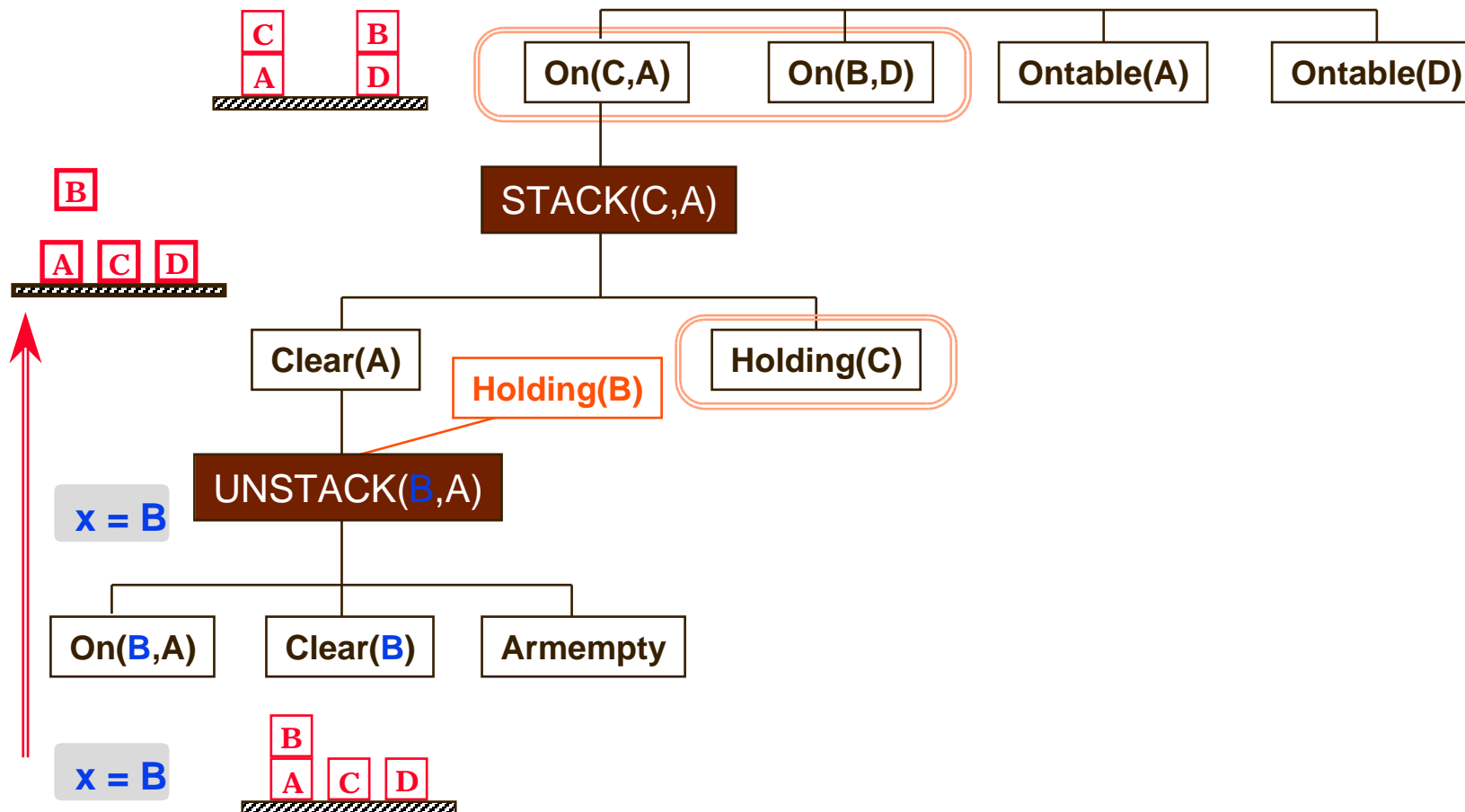
Exemplo



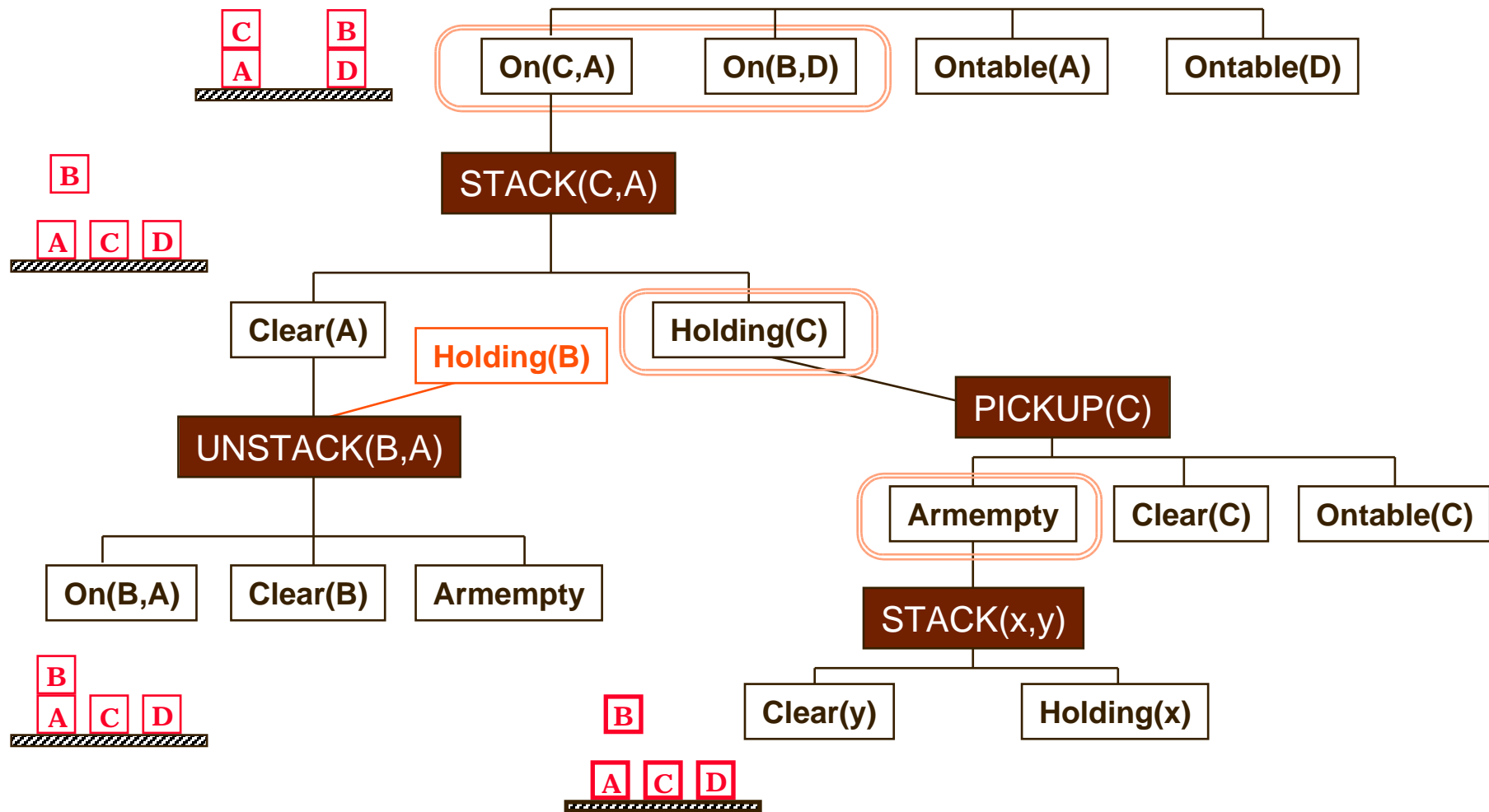
Exemplo



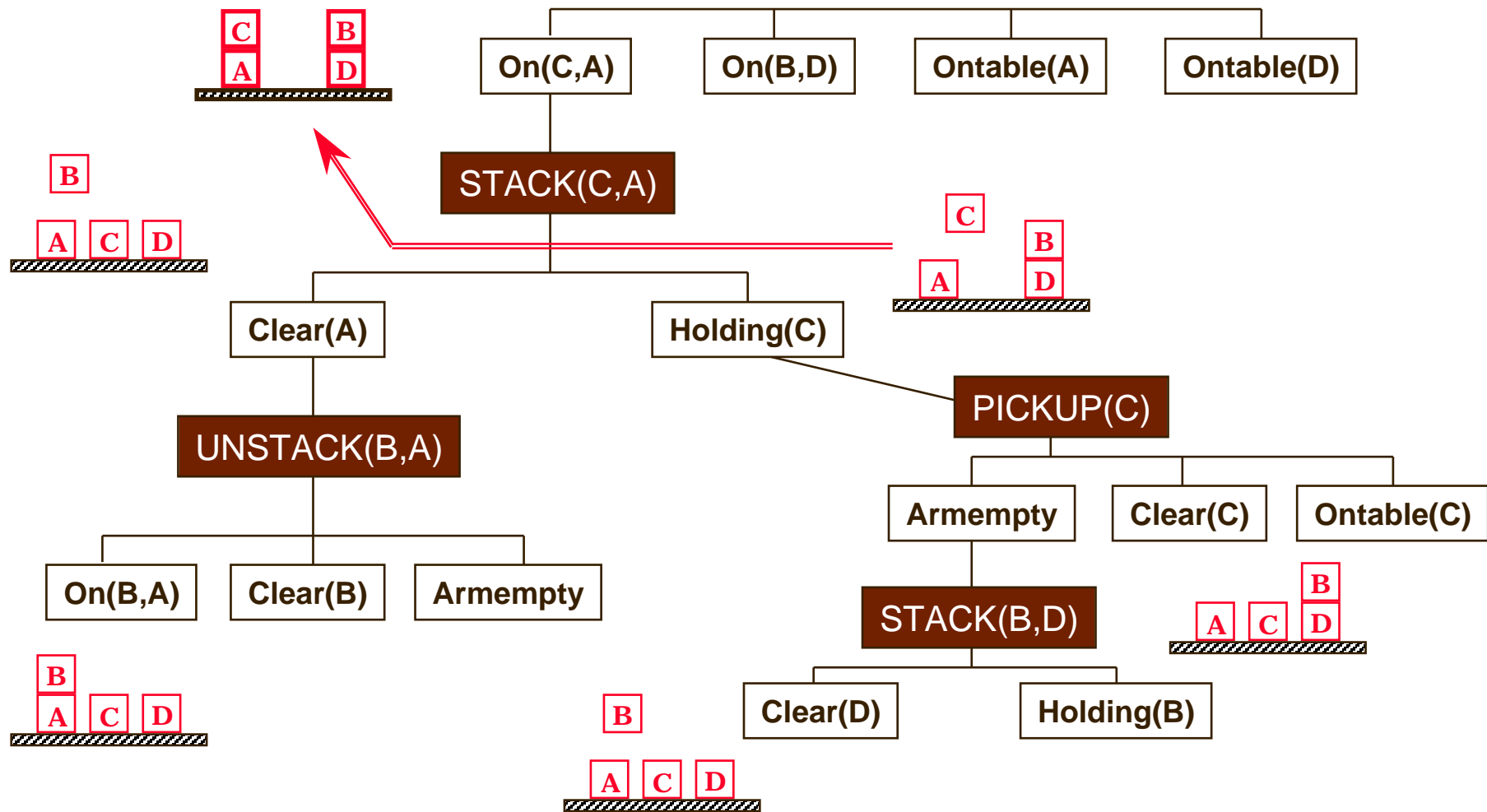
Exemplo



Exemplo



Exemplo



Assunção de Linearidade

Se os objectivos parciais (subgoals) são independentes, podem ser sequencialmente alcançados, independentemente da ordem (Sussman 1973)

- **A anomalia de Sussman:**

- **Problema:**



Assunção de Linearidade

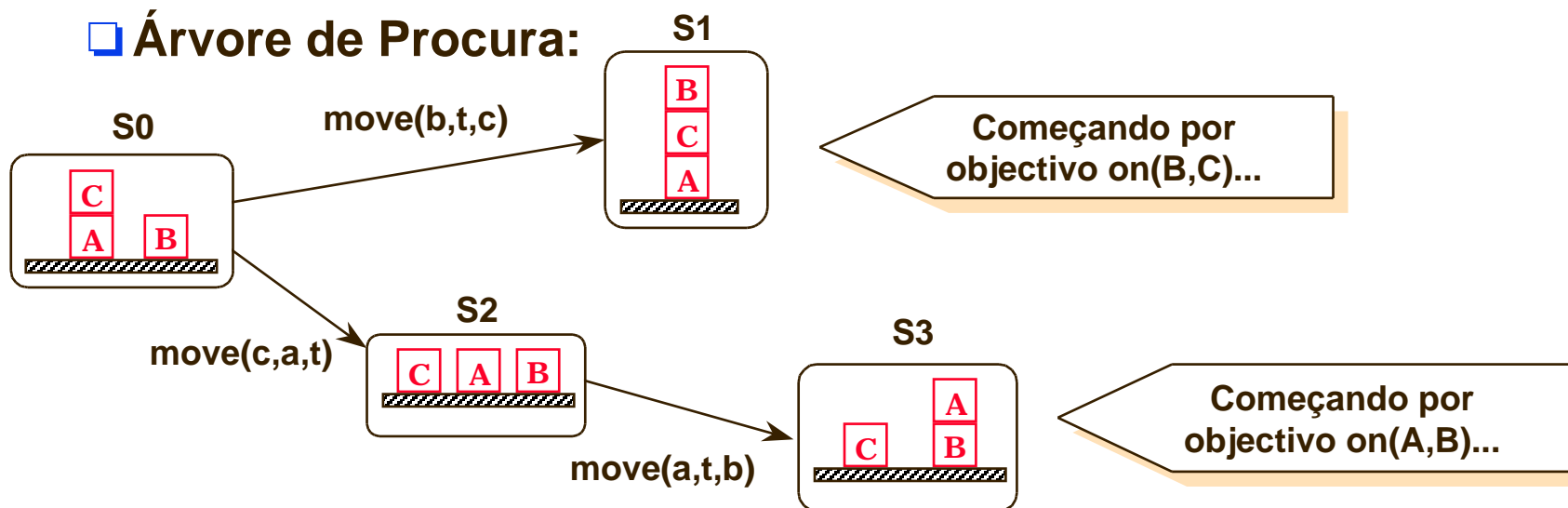
Se os objectivos parciais (subgoals) são independentes, podem ser sequencialmente alcançados, independentemente da ordem (Sussman 1973)

● A anomalia de Sussman:

□ Problema:



□ Árvore de Procura:



Outro exemplo

- Troca de valores entre registos:

