Geração Automática de planos

- Gerar planos é uma das atividades essenciais em qualquer sistema que pretenda ter um comportamento ``inteligente''.

- Planejar significa gerar um conjunto de ações que, quando executadas, mudam o estado da nossa aplicação de um estado inicial para um estado final, que é o nosso objetivo.

- Geradores de planos são parte fundamental de qualquer processo de automação, como por exemplo: automação industrial, robótica ou em sistemas de apoio a tomada de decisão.

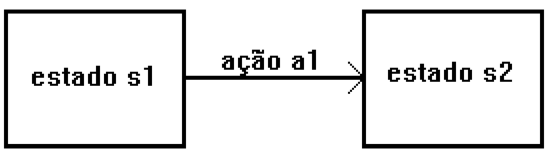
- Geração automática de planos tem interseção com várias áreas, como por exemplo: lógica clássica, lógica modal de ação, lógica temporal, lógicas não-monotônicas, provadores de teoremas, sistemas distribuídos, inteligência artificial, especificação de `software', controle e servomecanismos.

Problema de Planejamento

- um **mini-mundo** ou simplesmente um **mundo** é a parte do universo que nós queremos modelar. Por exemplo, uma mesa com alguns blocos de madeira sobre ela.

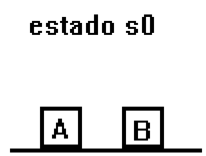
- um **estado** é a descrição do mundo em uma linguagem "formal"num determinado instante de tempo. Em geral, existe uma infinidade de estados em que o mundo pode estar, porém num instante determinado, ele pode estar em um único estado.

- uma **ação** é um operador que quando executado muda o estado do mundo. A única maneira do mundo mudar de estado é pela execução de alguma ação, caso contrário, ele permanece estático.

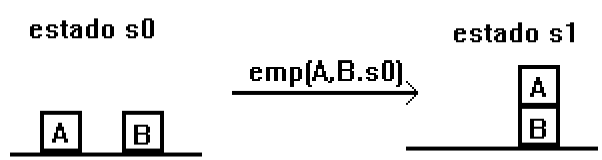


Mundo dos Blocos

- Blocos de madeira do mesmo tamanho, cada possuindo um rótulo diferente. Um bloco pode estar ou em cima da mesa ou sobre um outro bloco.



- Se a ação de empilhar o bloco A sobre o bloco B for executada, então o mundo vai mudar do estado s0 para o estado s1, como mostra a figura:



Definição: Um *problema de planejamento clássico* é uma quadupla < S, A , s0, Sg>, onde:

- S é o conjunto de todos os estados;
- A é o conjunto de ações;
- s0 é o estado inicial, $s0 \in S$;
- Sg é um conjunto de estados finais possíveis.

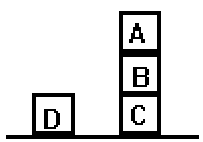
- A razão porque nós temos mais de um estado final é porque em geral o estado final é qualquer estado no qual uma certa propriedade seja veradadeira.

Cálculo de Situações

- Cálculo de situações é o uso de lógica de primeira ordem para representar problemas que envolvem ação e estado.

- Predicados são utilizados para representar propriedades sobre os estados.

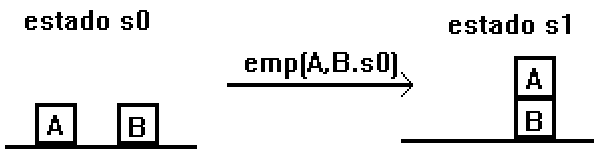
  - *Sobre(x,y,s)* - bloco x está sobre o bloco y no estado s;
  - *Limpo(x,s)* - não existe nenhum bloco sobre o bloco x no estado s;
  - *Mesa(x,s)* - bloco x está sobre a mesa no estado s.



Sobre(A,B,s1)	Sobre(B,C,s1)	Mesa(D,s1)
Limpo(A,s1)	Mesa(C,s1)	Limpo(D,s1)

- <u>Ações</u> são representadas por <u>funções</u> que mapeiam estados em estados, i.e., a: S→S.

  - *emp(x,y,s)* - empilha o bloco x sobre o bloco y no estados;
  - *desemp(x,y,s)* - desempilha o bloco x de cima do bloco y no estados;



Limpo(B,s0)	Limpo(A,s1)
Limpo(A,s0)	Mesa(B,s1)
Mesa(B,s0)	Sobre(A,B,s1)
Mesa(A,s0)	

- Estamos assumindo que nós temos uma lógica de primeira ordem com igualdade =, e que o símbolo ≠ é definido como ~= (não igual).

Grupo 1 de Axiomas: propiedades gerais dos estados.

1. $\forall x \forall s\ (Mesa(x,s) \leftrightarrow \sim\exists y\ (Sobre(x,y,s))$
2. $\forall y \forall s\ (Limpo(y,s) \leftrightarrow \sim\exists x\ (Sobre(x,y,s))$
3. $\forall x \forall y \forall z \forall s\ (Sobre(x,y,s) \rightarrow (Sobre(x,z,s) \leftrightarrow z=y))$
4. $\forall x \forall s\ (Limpo(x,s) \wedge Mesa(x,s)\ ) \rightarrow Livre(x,s)$

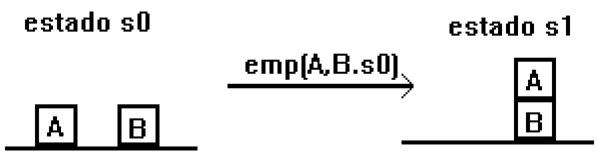
Grupo 2 de Axiomas: contém os axiomas que definem as ações, i.e., define em que condições uma ação pode ser executada (pré-condição) e como esta afeta o estado corrente.

5. $\forall x \forall y \forall s\ ((Mesa(x,s) \wedge Limpo(x,s) \wedge Limpo(y,s) \wedge x \neq y) \rightarrow Sobre(x,y,emp(x,y,s)))$

O axioma 5 descreve a ação de empilhar. Se um bloco x está sobre a mesa e limpo e um bloco y diferente de x está limpo,no estado s, então após executar a ação de empilhar x sobre y o bloco x estará sobre y no novo estado.

6. $\forall x \forall y \forall s\ ((\text{Sobre}(x,y,s) \wedge \text{Limpo}(x,s)) \rightarrow (\text{Mesa}(x,\text{desemp}(x,y,s)) \wedge \text{Limpo}(y,\text{desemp}(x,y,s))))$

O axioma 6 descreve a ação de desempilhar. Se um bloco x está sobre y e x está limpo em s, então após executar a ação de desempilhar x de cima de y o bloco x estará sobre a mesa e y limpo no novo estado.



Limpo(B,s0)	Limpo(A,s1)
Limpo(A,s0)	Mesa(B,s1)
Mesa(B,s0)	Sobre(A,B,s1)
Mesa(A,s0)	

- Como concluir Limpo(A,s1) e Mesa(B,s1)?
- Como trazer propriedades que não foram afetadas pela execução da ação para o novo estado?

Grupo 3 de Axiomas: contém os axiomas de "frame". Eles descrevem que propriedades não foram afetadas quando uma certa ação foi executada. Os axiomas do grupo 2 descrevem como uma ação muda o estado, mas eles não dizem como as propriedades que não foram afetadas podem passar para o novo estado. Isto é o que fazem os axiomas de frame.

Axiomas de frame para a ação de empilha:

7. $\forall x \forall y \forall u \forall s\ ((Limpo(u,s) \land\ u \neq y) \rightarrow Limpo(u,emp(x,y,s)))$
8. $\forall x \forall y \forall u \forall s\ ((Mesa(u,s) \land\ u \neq x) \rightarrow Mesa(u,emp(x,y,s)))$
9. $\forall x \forall y \forall u \forall v \forall s\ (Sobre(u,v,s) \rightarrow Sobre(u,v,emp(x,y,s)))$

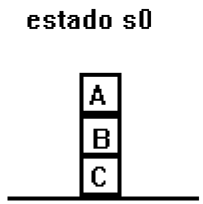
Axiomas 7,8 e 9 dizem que se um bloco u está limpo, ou sobre a mesa ou sobre um bloco v, e se um bloco x é empilhado sobre um bloco y então u permanece na mesma situação, i.e., limpo ou sobre a mesa ou sobre v.

Axiomas de frame para a ação de desempilha:

10. $\forall x \forall y \forall u \forall s\ (Limpo(u,s) \rightarrow Limpo(u,desemp(x,y,s)))$
11. $\forall x \forall y \forall u \forall s\ (Mesa(u,s) \rightarrow Mesa(u,desemp(x,y,s)))$
12. $\forall x \forall y \forall u \forall v \forall s\ ((Sobre(u,v,s) \wedge u \neq x) \rightarrow Sobre(u,v,desemp(x,y,s)))$

Axiomas 10,11 e 12 dizem que se um bloco u está limpo, ou sobre a mesa ou sobre um bloco v, e se um bloco x é desempilhado de cima de um bloco y então u permanece na mesma situação, i.e., limpo ou sobre a mesa ou sobre v.

Exemplo:



- Descrição do estado inicial:

F1. Sobre(A,B,s0) F2. Limpo(A,s0)
F3. Sobre(B,C,s0) F4. Mesa(C,s0)

- O objetivo é atingir um estado onde o bloco B esteja livre Livre(B).
- Aplicando axioma 6 sobre os fatos F1 e F2:

1. Mesa(A,desempilha(A,B,s0))
1'. Limpo(B,desempilha(A,B,s0)

- Usando o axioma de frame 12 e fato F3:

2. Sobre(B,C,desempilha(A,B,s0))

- Usando axioma 6 outra vez e 2 e 1':

3. Mesa(B,desempilha(B,C,desempilha(A,B,s0)))
3'. Limpo(C,desempilha(B,C,desempilha(A,B,s0)))

- Usando o axioma de frame 10 e fato 1':

4. Limpo(B,desempilha(B,C,desempilha(A,B,s0)))

- Usando axioma 4:

5. Livre(B,desempilha(B,C,desempilha(A,B,s0)))

- Vantagem Cálculo de Situações:

    - baseado em lógica de primeira ordem;
    - formal;
    - provadores de teoremas disponíveis no mercado;

- Desvantagem:

    - Número de axiomas de frame é proporcional ao número de predicados vezes o número de ações. Pode ser muito grande.

    - Propriedades que não são afetadas pelas ações tem que ser deduzidas no novo estado usando-se os axiomas de frame, isto faz com que as deduções fiquem bastante longas e ineficientes.