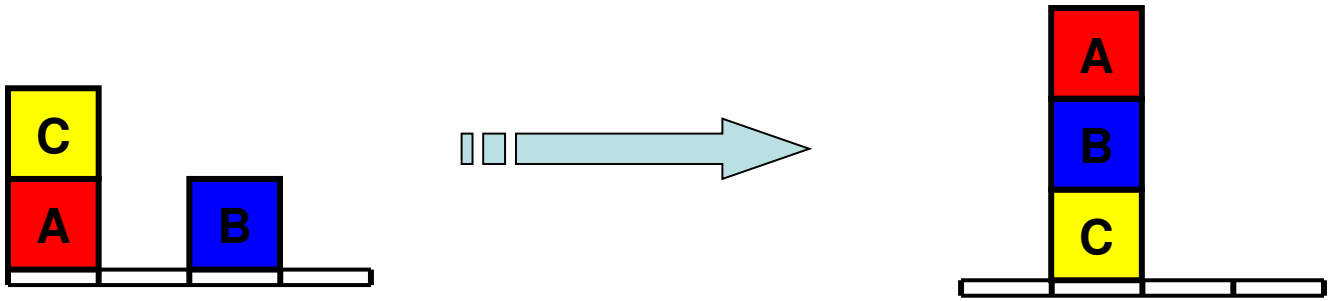
Planning Problems

- Want a sequence of actions to turn a start state into a goal state

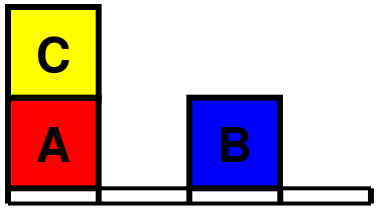


- Unlike generic search, states and actions have internal structure, which allows better search methods

This slide deck courtesy of Dan Klein at UC Berkeley

State Space

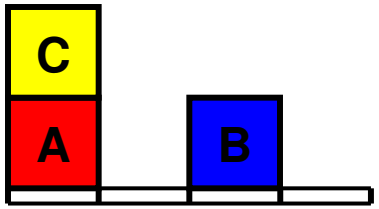
On(C, A)
On(A, Table)
On(B, Table)
Clear(C)
Clear(B)



- Representation
  - States described by propositions or ground predicates
  - Sparse encoding (database semantics): all unstated literals are false
  - Unique names: each object has its own single symbol

Actions

On(C, A)
On(A, Table)
On(B, Table)
Clear(C)
Clear(B)

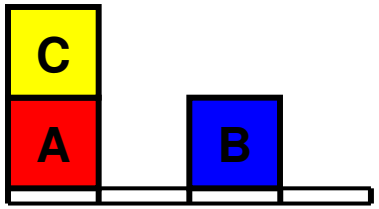


ACTION: Move(b,x,y)
PRECONDITIONS: On(b,x), Clear(b), Clear(y)
POSTCONDITIONS: On(b,y), Clear(x)
¬On(b,x), ¬Clear(y)

ACTION: Move(C,A,Table)
PRECONDITIONS: On(C,A), Clear(C),
Clear(Table)
POSTCONDITIONS: On(C,Table), Clear(A)
¬On(C,A), ¬Clear(Table)

Actions

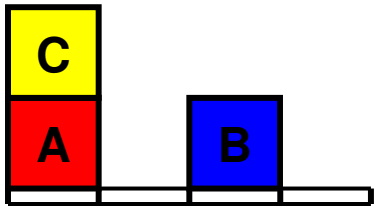
On(C, A)
On(A, Table)
On(B, Table)
Clear(C)
Clear(B)



ACTION: MoveToBlock(b,x,y)
PRECONDITIONS: On(b,x), Clear(b), Clear(y),
Block(b), Block(y), (b≠ x), (b≠ y), (x≠ y)
POSTCONDITIONS: On(b,y), Clear(x)
¬On(b,x), ¬Clear(y)

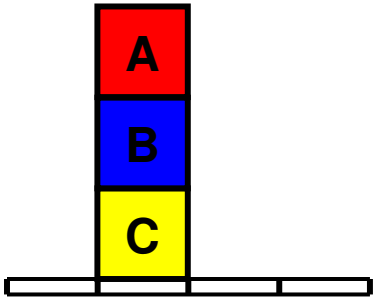
ACTION: MoveToTable(b,x)
PRECONDITIONS: On(b,x), Clear(b), Block(b), Block(x), (b≠ x)
POSTCONDITIONS: On(b,Table), Clear(x)
¬On(b,x)

Start and Goal States



Start State

On(C, A)
On(A, Table)
On(B, Table)
Clear(C)
Clear(B)
Block(A)
…



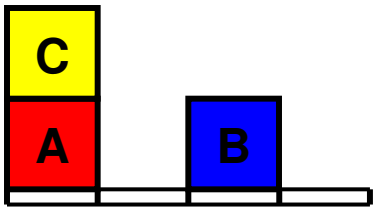
Goal State

On(B, C)
On(A, B)

Important: goal satisfied by any state which entails goal list

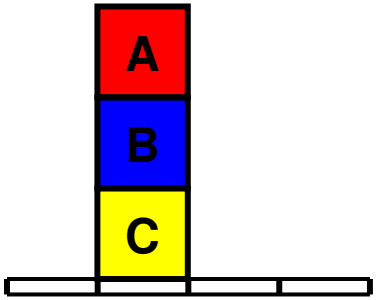
[MoveToTable(C,A), Move(B,Table,C), Move(A,Table,B)]

Planning Problems



On(C, A)
On(A, Table)
On(B, Table)
Clear(C)
Clear(B)

Sparse encoding, but complete state spec



Goal

Set of goal states, only requirements specified (think unary constraints)

On(B, C)
On(A, B)

Action schema, instantiates to give specific ground actions

Which goal first?

ACTION: MoveToTable(b,x)
PRECONDITIONS: On(b,x), Clear(b), Block(b), Block(x), (b≠ x)
POSTCONDITIONS: On(b,Table), Clear(x)
¬On(b,x)

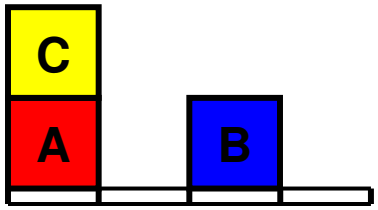
Practice

- Problem 10.2: “Applicable”
- Problem 10.3a,b: Representation

  Where do they come from?

  Could they be learned?

Kinds of Plans



Start State

On(C, A)
On(A, Table)
On(B, Table)
Clear(C)
Clear(B)
Block(A)
…

Sequential Plan

MoveToTable(C,A) > Move(B,Table,C) > Move(A,Table,B)

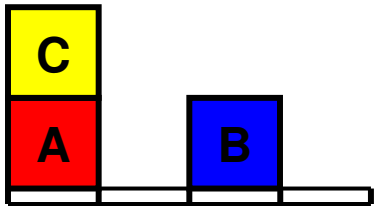
Partial-Order Plan

MoveToTable(C,A)

Move(B,Table,C)

> Move(A,Table,B)]

Forward Search



Start State

On(C, A)
On(A, Table)
On(B, Table)
Clear(C)
Clear(B)
Block(A)
…

MoveToTable(C,A)

MoveToBlock(C,A,B)

MoveToBlock(B,Table,C)

~~On(C, A)~~
On(A, Table)
On(B, Table)
Clear(C)
Clear(B)
Block(A)
…
+Clear(A)
+On(C, Table)

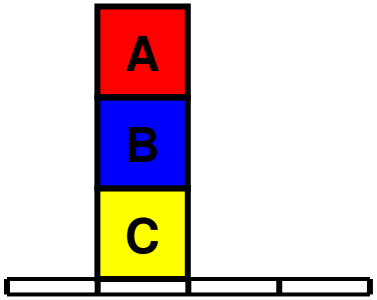
Applicable actions

Backward Search

ACTION: MoveToBlock(b,x,y)
PRECONDITIONS: On(b,x), Clear(b), Clear(y),
Block(b), Block(y), (b≠ x), (b≠ y),
(x≠ y)
POSTCONDITIONS: On(b,y), Clear(x)
¬On(b,x), ¬Clear(y)

MoveToBlock(A,Table,B)

MoveToBlock(A,x',B)



Goal State

On(B, C)
On(A, B)

On(B, C)
~~On(A, B)~~
+On(A,Table)
+Clear(A)
+Clear(B)
+…

Relevant actions

$$g' = (g - \text{ADD}(a)) \cup Precond(a)$$