Planning and Search

Classical Planning 2

Outline

◊ STRIPS planning algorithm: goal stack planning

◊ Example: Sussman anomaly

An example domain: the blocks world

The domain consists of:

1. a table, a set of cubic blocks and a robot arm;
2. each block is either on the table or stacked on top of another block
3. the arm can pick up a block and move it to another position either on the table or on top of another block;
4. the arm can only pick up one block at time, so it cannot pick up a block which has another block on top.

A **goal** is a request to build one or more stacks of blocks specified in terms of what blocks are on top of what other blocks.

State descriptions

Blocks are represented by constants $A, B, C \ldots$ etc. and an additional constant $Table$ representing the table.

The following predicates are used to describe states:

On(b, x)	block b is on x, where x is either another block or the table
Clear(x)	there is a clear space on x to hold a block

Actions schemas

Actions schemas:

MOVE(b, x, y):
PRECOND: $\mathit{On}(b, x) \wedge \mathit{Clear}(b) \wedge \mathit{Clear}(y)$
EFFECT: $\mathit{On}(b, y) \wedge \mathit{Clear}(x) \wedge$
$\neg \mathit{On}(b, x) \wedge \neg \mathit{Clear}(y)$

MOVE-TO-TABLE(b, x):
PRECOND: $\mathit{On}(b, x) \wedge \mathit{Clear}(b)$
EFFECT: $\mathit{On}(b, \mathit{Table}) \wedge \mathit{Clear}(x) \wedge$
$\neg \mathit{On}(b, x)$

Some problems with this formalisation

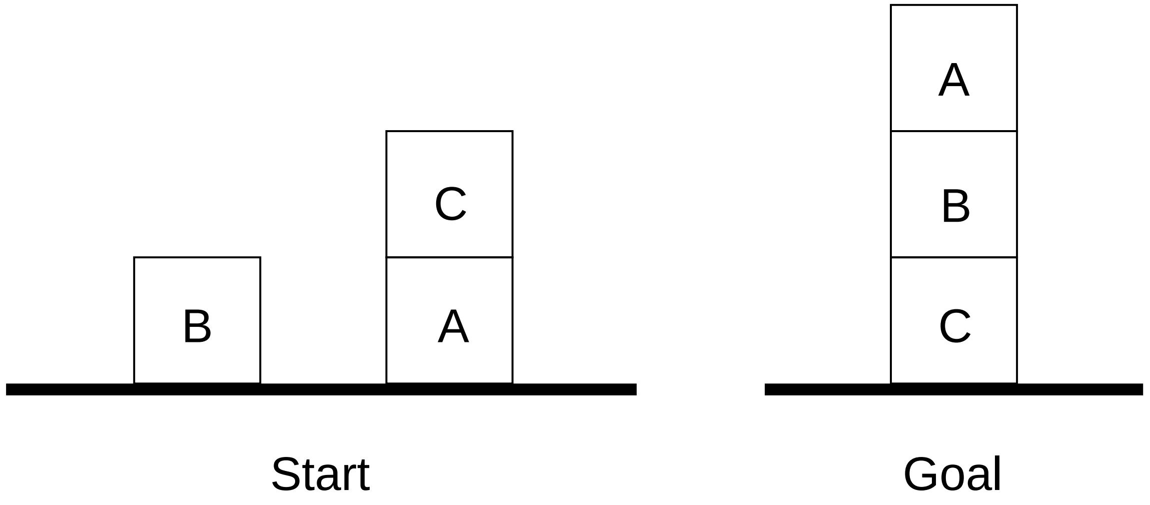
There is nothing to stop the planner using the ‘wrong’ operator to put a block on the table, i.e. $\mathrm{MOVE}(b, x, \textit{Table})$ rather than $\mathrm{MOVE\text{-}TO\text{-}TABLE}(b, x)$ which results in a larger-than-necessary search space; and

Some of the operator applications, such as $\mathrm{MOVE}(B, C, C)$, which should do nothing, have inconsistent effects

To solve the first one, we could introduce predicate $Block$ and make $Block(x)$ and $Block(y)$ preconditions of $\mathrm{MOVE}(x, y, z)$

To solve the second we could add $\neg(y = z)$ as a precondition for $\mathrm{MOVE}(x, y, z)$

An example problem: Sussman anomaly



Initial state

$$\mathit{On}(C, A) \wedge \mathit{OnTable}(A) \wedge \mathit{OnTable}(B) \wedge Clear(B) \wedge Clear(C)$$

Goal

$$\mathit{On}(A, B) \wedge \mathit{On}(B, C) \wedge \mathit{OnTable}(C)$$

Goal stack planning

One of the earlier planning algorithms called goal stack planning. It was used by STRIPS.

We work backwards from the goal, looking for an operator which has one or more of the goal literals as one of its effects and then trying to satisfy the preconditions of the operator. The preconditions of the operator become subgoals that must be satisfied. We keep doing this until we reach the initial state.

Goal stack planning uses a stack to hold goals and actions to satisfy the goals, and a knowledge base to hold the current state, action schemas and domain axioms

Goal stack is like a node in a search tree; if there is a choice of action, we create branches

Goal stack planning pseudocode

Push the original goal on the stack. Repeat until the stack is empty:

If stack top is a compound goal, push its unsatisfied subgoals on the stack.

If stack top is a single unsatisfied goal, replace it by an action that makes it satisfied and push the action's precondition on the stack.

If stack top is an action, pop it from the stack, execute it and change the knowledge base by the action's effects.

If stack top is a satisfied goal, pop it from the stack.

Goal stack planning 1

(I do pushing the subgoals at the same step as the compound goal.) The order of subgoals is up to us; we could have put *On*(B, C) on top

> *On*(A, B)
> *On*(B, C)
> *OnTable*(C)
> *On*$(A, B) \wedge$ *On*$(B, C) \wedge$ *OnTable*(C)

$$KB = \{On(C, A), OnTable(A), OnTable(B), Clear(B), Clear(C)\}$$

plan $= [\,]$

The top of the stack is a single unsatisfied goal. We push the action which would achieve it, and its preconditions.

Goal stack planning 2

$\mathit{On}(A, Table)$
$\mathit{Clear}(B)$
$\mathit{Clear}(A)$
$\mathit{On}(A, Table) \wedge \mathit{Clear}(A) \wedge \mathit{Clear}(B)$
$\mathrm{MOVE}(A, x, B)$
$\mathit{On}(A, B)$
$\mathit{On}(B, C)$
$\mathit{OnTable}(C)$
$\mathit{On}(A, B) \wedge \mathit{On}(B, C) \wedge \mathit{OnTable}(C)$

$KB = \{On(C, A), OnTable(A), OnTable(B), Clear(B), Clear(C)\}$

plan $= [\,]$

The top of the stack is a satisfied goal. We pop the stack (twice).

Goal stack planning 3

$\mathit{Clear}(A)$
$\mathit{On}(A, Table) \wedge \mathit{Clear}(A) \wedge \mathit{Clear}(B)$
$\mathrm{MOVE}(A, Table, B)$
$\mathit{On}(A, B)$
$\mathit{On}(B, C)$
$\mathit{OnTable}(C)$
$\mathit{On}(A, B) \wedge \mathit{On}(B, C) \wedge \mathit{OnTable}(C)$

$KB = \{On(C, A), OnTable(A), OnTable(B), Clear(B), Clear(C)\}$

plan $= [\,]$

The top of the stack is an unsatisfied goal. We push the action which would achieve it, and its preconditions.

Goal stack planning 4

$\mathit{On}(C, A)$
$\mathit{Clear}(C)$
$\mathit{On}(C, A) \wedge \mathit{Clear}(C)$
$\text{MOVE-TO-TABLE}(C, A)$
$\mathit{Clear}(A)$
$\mathit{On}(A, Table) \wedge \mathit{Clear}(A) \wedge \mathit{Clear}(B)$
$\text{MOVE}(A, Table, B)$
$\mathit{On}(A, B)$
$\mathit{On}(B, C)$
$\mathit{OnTable}(C)$
$\mathit{On}(A, B) \wedge \mathit{On}(B, C) \wedge \mathit{OnTable}(C)$

$KB = \{On(C, A), OnTable(A), OnTable(B), Clear(B), Clear(C)\}$

plan $= [\ ]$

The top of the stack is a satisfied goal. We pop the stack (three times).

Goal stack planning 5

MOVE-TO-TABLE(C, A)
Clear(A)
On$(A, Table) \wedge$ *Clear*$(A) \wedge$ *Clear*(B)
MOVE$(A, Table, B)$
On(A, B)
On(B, C)
OnTable(C)
On$(A, B) \wedge$ *On*$(B, C) \wedge$ *OnTable*(C)

$KB = \{On(C, A), OnTable(A), OnTable(B), Clear(B), Clear(C)\}$

plan $= [\,]$

The top of the stack is an action. We execute it, update the KB with its effects, and add it to the plan.

Goal stack planning 6

$\mathit{Clear}(A)$
$\mathit{On}(A, Table) \wedge \mathit{Clear}(A) \wedge \mathit{Clear}(B)$
$\mathrm{MOVE}(A, Table, B)$
$\mathit{On}(A, B)$
$\mathit{On}(B, C)$
$\mathit{OnTable}(C)$
$\mathit{On}(A, B) \wedge \mathit{On}(B, C) \wedge \mathit{OnTable}(C)$

$KB = \{OnTable(C), OnTable(A), OnTable(B), Clear(A), Clear(B), Clear(C)\}$

plan $= [\text{MOVE-TO-TABLE}(C, A)]$

The top of the stack is a satisfied goal. We pop the stack (twice).

Goal stack planning 7

$\text{MOVE}(A, Table, B)$
On(A, B)
On(B, C)
OnTable(C)
On$(A, B) \wedge$ *On*$(B, C) \wedge$ *OnTable*(C)

$KB = \{OnTable(C), OnTable(A), OnTable(B), Clear(A), Clear(B), Clear(C)\}$

plan $= [\text{MOVE-TO-TABLE}(C, A)]$

The top of the stack is an action. We execute it, update the KB with its effects, and add it to the plan.

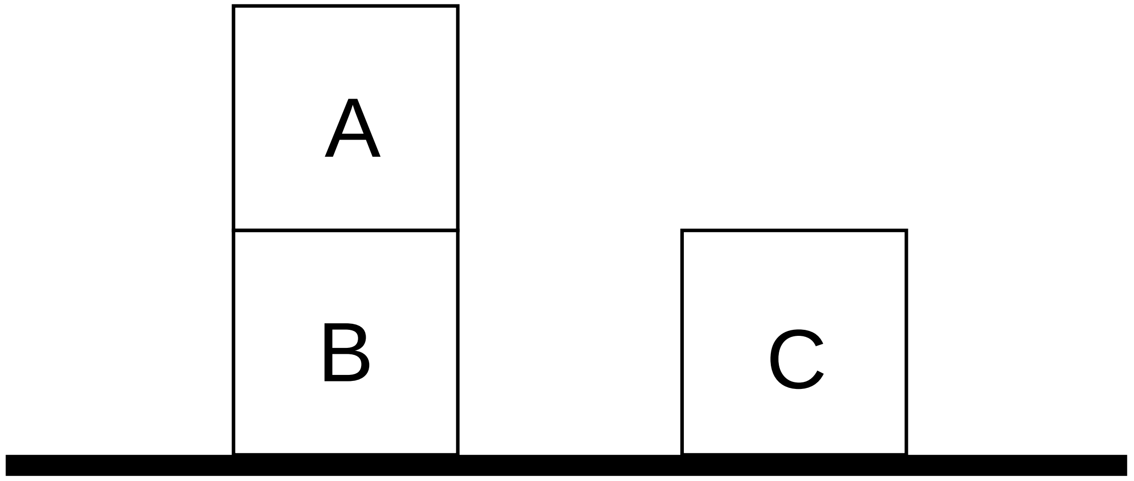
Goal stack planning 8

$\mathit{On}(A,B)$
$\mathit{On}(B,C)$
$\mathit{OnTable}(C)$
$\mathit{On}(A,B) \wedge \mathit{On}(B,C) \wedge \mathit{OnTable}(C)$

$KB = \{\mathit{On}(A,B), \mathit{OnTable}(C), \mathit{OnTable}(B), \mathit{Clear}(A), \mathit{Clear}(C)\}$

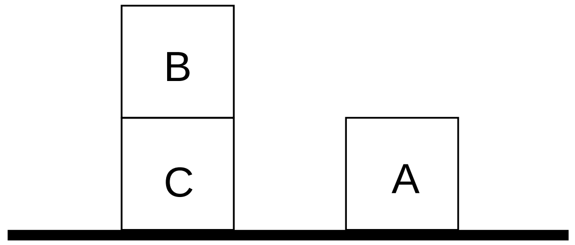
plan $= [\mathrm{MOVE\text{-}TO\text{-}TABLE}(C,A), \mathrm{MOVE}(A, Table, B)]$

the current state is



Goal stack planning 9

If we follow the same process for the *On*(B, C) goal, we end up in the state



$$\textit{On}(B, C) \wedge \textit{OnTable}(A) \wedge \textit{OnTable}(C)$$

The remaining goal on the stack,

$$\textit{On}(A, B) \wedge \textit{On}(B, C) \wedge \textit{OnTable}(C)$$

is not satisfied. So *On*(A, B) will be pushed on the stack again.

Goal stack planning 10

Now we finally can move A on top of B, but the resulting plan is redundant:

> MOVE-TO-TABLE(C, A)
> MOVE$(A, Table, B)$
> MOVE-TO-TABLE(A, B)
> MOVE$(B, Table, C)$
> MOVE$(A, Table, B)$

There are techniques for ‘fixing’ inefficient plans (where something is done and then undone) but it is difficult in general (when it is not straight one after another)

Why this is an instructive example

Sussman anomaly (called ‘anomaly’ because it seemed to make sense for a conjunctive goals to first achieve one goal and then achieve another goal, and then the complete goal would be achieved) is instructive, because achieving one goal ($\mathit{On}(A, B)$) destroys preconditions of an action which is necessary to achieve the other goal ($\mathit{On}(B, C)$), namely $\mathit{Clear}(B)$.

Such interaction between actions is called clobbering

More in the next lecture

Next lecture

Partial order planning: Russell and Norvig, 3rd ed., 10.4.4.

This lecture was based on Elaine Rich and Kevin Knight, Artificial Intelligence textbook.