

Intelligent Agents

State-Space Search

Ute Schmid

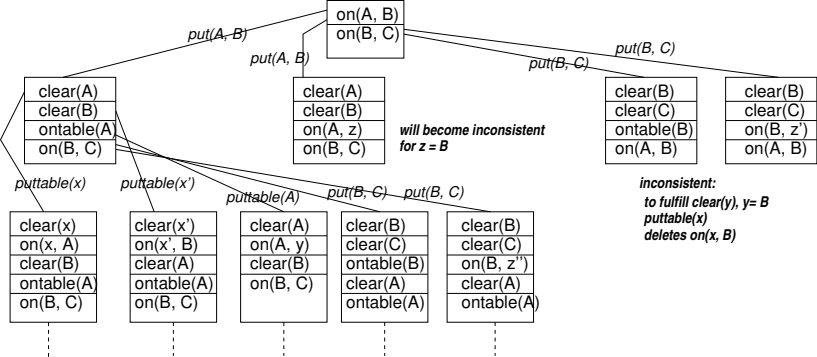
Cognitive Systems, Applied Computer Science, Bamberg University

last change: 27. Mai 2010

Remarks on Backward Planning

- Also called **regression planning**
- Advantage: typically smaller search trees
- Problem: inconsistent states can be produced
can e.g. be detected by including axioms (domain knowledge!)
- Graphplan strategy: build a Planning Graph by forwards search (polynomial effort) and extract the plan from the graph backwards (exponential effort, as usual for planning)

Backward Planning cont.



STRIPS

- by Fikes & Nilsson (1971), “**Stanford Research Institute Problem Solver**”
- classical example: moving boxes between rooms (“Strips World”)
- Originally: representation formalism (relying on CWA) and planning algorithm
today: “STRIPS planning” refers to classical representation without extensions and not to a specific algorithm
- STRIPS algorithm: a **linear** (and therefore incomplete) approach
- compare to: General Problem Solver (GPS), a cognitively motivated problem solving algorithm which is also linear and therefore incomplete

STRIPS Algorithm

- Backward-search with a kind of hill climbing strategy
- In each recursive call only such subgoals are relevant which are preconditions of the last operator added
- Consequence: considerable reduction of branching, but resulting in incompleteness
- Linear planning: organizing subgoals in a stack
- Non-linear planning: organizing subgoals in a set, interleaving of goals

STRIPS Algorithm

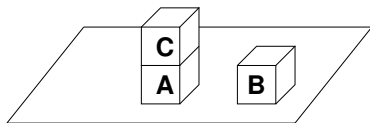
STRIPS(O, s, g)

- $\Pi \leftarrow$ empty plan
- loop
 - ▶ if s satisfies g then return Π
 - ▶ $A \leftarrow \{a \mid a \text{ is a ground instance of an operator in } O, \text{ and } a \text{ is relevant for } g\}$
 - ▶ if $A = \emptyset$ then return failure
 - ▶ nondeterministically choose any action $a \in A$
 - ▶ $\Pi' \leftarrow \text{STRIPS}(O, s, \text{precond}(a))$
 - ▶ if $\Pi' = \text{failure}$ then return failure
;; if we get here, then Π' achieves $\text{precond}(a)$ from s
 - ▶ $s \leftarrow \gamma(s, \Pi')$;; s now satisfies $\text{precond}(a)$
 - ▶ $s \leftarrow \gamma(s, a)$
 - ▶ $\Pi \leftarrow \Pi.\Pi'.a$

Incompleteness of Linear Planning

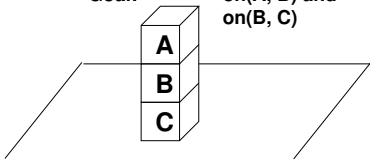
The Sussman Anomaly

Initial State

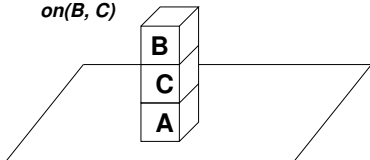


Goal:

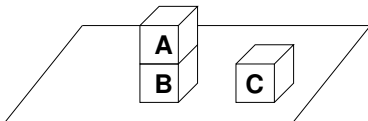
$on(A, B)$ and
 $on(B, C)$



$on(B, C)$



$on(A, B)$



Sussman Anomaly

Linear planning corresponds to dealing with goals organized in a **stack**:

$[on(A, B), on(B, C)]$

try to satisfy goal $on(A, B)$


 solve sub-goals $[clear(A), clear(B)]$ ¹

 all sub-goals hold after $puttable(C)$

 apply $put(A, B)$

goal $on(A, B)$ is reached

try to satisfy goal $on(B, C)$.

¹We ignore the additional subgoal $ontable(A)$ resp. $on(A, z)$ here. 

Interleaving of Goals

- Non-linear planning allows that a sequence of planning steps dealing with one goal is interrupted to deal with another goal.
- For the Sussman Anomaly, that means that after block C is put on the table pursuing goal $on(A, B)$, the planner switches to the goal $on(B, C)$.
- Non-linear planning corresponds to dealing with goals organized in a **set**.
- The correct sequence of goals might not be found immediately without backtracking.

Interleaving of Goals cont.

$\{on(A, B), on(B, C)\}$

try to satisfy goal $on(A, B)$

$\{clear(A), clear(B), on(A, B), on(B, C)\}$

$clear(A)$ and $clear(B)$ hold after $puttable(C)$

try to satisfy goal $on(B, C)$

apply $put(B, C)$

try to satisfy goal $on(A, B)$

apply $put(A, B)$.

Rocket Domain

(Veloso)

- Objects: n boxes, Positions (Earth, Moon), one Rocket
- Operators: load/unload a box, move the Rocket (oneway: only from earth to moon, no way back!)
- Linear planning: to reach the goal, that Box1 is on the Moon, load it, shoot the Rocket, unload it, now no other Box can be transported!