Tutorial for November 17, 21

See Russell and Norvig, chapter 11.

function POP(*initial, goal, operators*) **returns** *plan*

plan ← MAKE-MINIMAL-PLAN(*initial, goal*)
loop do
  if SOLUTION?(*plan*) **then return** *plan*
  S_{need}, c ← SELECT-SUBGOAL(*plan*)
  CHOOSE-OPERATOR(*plan, operators*, S_{need}, c)
  RESOLVE-THREATS(*plan*)
end

---

function SELECT-SUBGOAL(*plan*) **returns** S_{need}, c

pick a plan step S_{need} from STEPS(*plan*)
  with a precondition c that has not been achieved
return S_{need}, c

---

procedure CHOOSE-OPERATOR(*plan, operators*, S_{need}, c)

choose a step S_{add} from *operators* or STEPS(*plan*) that has c as an effect
if there is no such step **then fail**
add the causal link $S_{add} \xrightarrow{c} S_{need}$ to LINKS(*plan*)
add the ordering constraint $S_{add} \prec S_{need}$ to ORDERINGS(*plan*)
if S_{add} is a newly added step from *operators* **then**
  add S_{add} to STEPS(*plan*)
  add $Start \prec S_{add} \prec Finish$ to ORDERINGS(*plan*)

---

procedure RESOLVE-THREATS(*plan*)

for each S_{threat} that threatens a link $S_i \xrightarrow{c} S_j$ in LINKS(*plan*) **do**
  choose either
    Demotion: Add $S_{threat} \prec S_i$ to ORDERINGS(*plan*)
    Promotion: Add $S_j \prec S_{threat}$ to ORDERINGS(*plan*)
  if not CONSISTENT(*plan*) **then fail**
end

Figure 1: POP algorithm

1. Consider forming a plan to put on socks and shoes, assuming that there are two identical socks, a left show and a right shoe; the goal is to have two shoes on, and a precondition to putting a shoe on should be that there is a sock on the foot in question.

   (a) Specify action operators for putting on socks, left shoe, right shoe by giving names for the actions, and associating preconditions and effects with the actions. The preconditions and effects should be (conjunction of) some (possibly negated) atomic statements.

(b) Use the partial order planning algorithm in figure 1 to work out a partial-order plan for realising the goal of having both shoes on. In this case, none of the links in the plan are threatened.

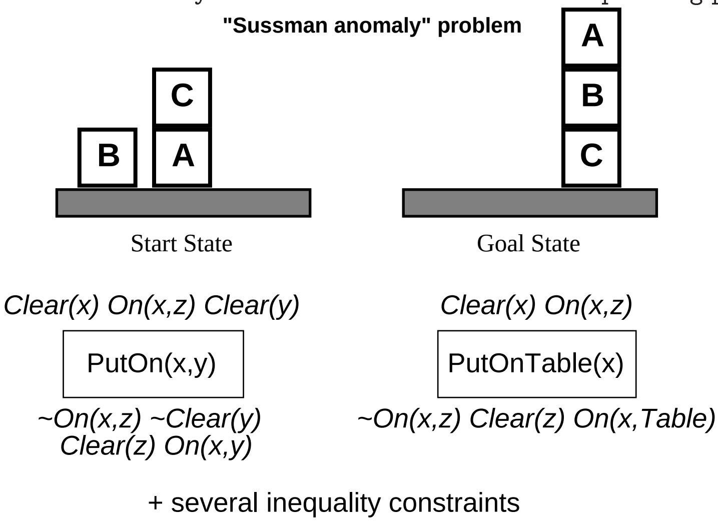
(c) How many ways are there of executing the plan linearly (one action after another)?

2. Formulate the following as a planning problem, and find a plan to solve it.

   You have two pots, one with capacity 5 litres, full of water and one with capacity two litres, empty. The goal is to have the two litre pot containing 1 litre of water.

3. Sketch a forward-chaining algorithm that uses planning operators as above to search for linear plans (sequences of actions) that may achieve the desied goal. You should aim for a **complete** algorithm, i.e. if there is a plan that will achieve the goal, your algorithm should achieve the plan.

4. The Sussman anomoly is based on a small blocks world planning problem.



   What are the non-redundant plans to achieve *On(A,B)*, *On(B,C)* as separate goals, independently from the starting state?

   Can these two plans be interleaved to from a plan that will achieve **both** goals?