Lab #3. Planning Agent in the blocks world

The figure shows a blocks-world planning problem known as the **Sussman anomaly.** The problem was considered anomalous because the noninterleaved planners could not solve it. Encode the problem in Java using STRIPS operators, and use Partial Order Planning (POP) to solve it.

Start State Goal State

Hints: STRIPS (STanford Research Institute Problem Solver) language, as described in the textbook, is commonly used in planning, and deals with operators that consist of prerequisites, add lists, and delete lists. Planning ultimately is a kind of searching for operator sequences that lead to a goal situation (a collection of desired assertions).
The initial situation of three blocks is:

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

And the goal situation is: On(A, B) ^ On(B, C)

First we'll explain why it is an anomaly for a noninterleaved planner. There are two subgoals; suppose we decide to work on On(A, B) first.

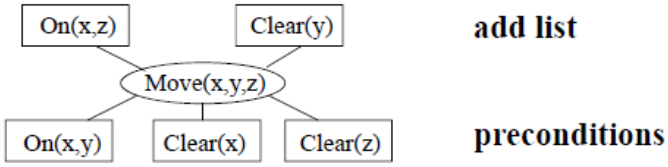
Initial state

We can clear C off A and then move A on to B. But then there is no way to achieve On(B, C)without undoing the work we have done.

Similarly, if we work on the subgoal On(B, C) first we can immediately achieve it in one step, but then we have to undo it to get A on B.

Initial state

We'll show next how things work out with an interleaved planner such as POP also called nonlinear planner. Since On(A, B) isn't true in the initial state, there is only one way to achieve it: Move(A, x, B), for some x.



Similarly, we also need Move(B, x', C) step for some x'. Now let's look at the Move(A, x, B) step. We need to achieve its precondition Clear(A). We could do that either with Move(b, A, y) or with MoveToTable(b, A). Let's assume we choose the latter. Now if we bind b to C, then all of the preconditions for the step MoveToTable(C, A) are true in the initial state, and we can add causal links to them. We then notice that there is a threat: the Move(B, x', C) step threatens the Clear(C) condition that is required by the MoveToTable step. We can resolve the threat by ordering Move(B, x', C) after the MoveToTable step. Finally, notice that all the preconditions for Move(B, x', C) are true in the initial state. Thus, we have a complete plan with all the preconditions satisfied. It turns out there is a well-ordering of the three steps:

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