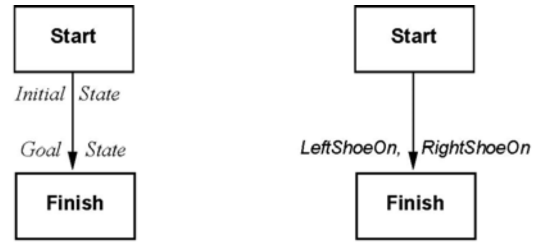
Definition of a Partially-Ordered Plan

- **A set of plan steps (actions).**
- **A set of step ordering constraints of the form**
  $S_i \prec S_j$ **written as** $S_i \longrightarrow S_j$
- **A set of variable binding constraints**
- **A set of causal links, written as**

$$S_i \xrightarrow{c} S_j$$

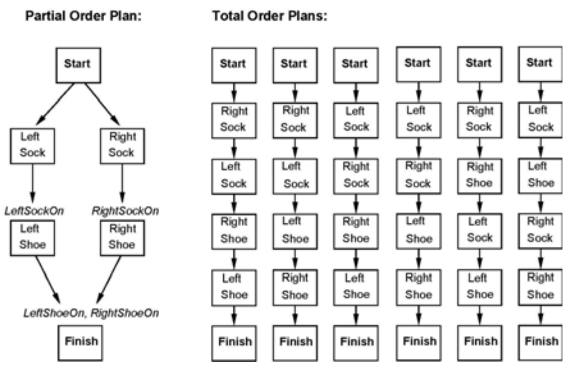
Initial Plan for Shoes and Socks

Initial plan: $Start \prec Finish$



(a) (b)

Partial Plan for Shoes and Socks



Planner Output

A solution is a complete, consistent plan.

1. **A complete plan:** every precondition of every step is achieved by some other step.
2. **A consistent plan:** there are no contradictions in the ordering or causal constraints. Contradiction occurs when both $S_i \prec S_j$ and $S_j \prec S_i$, or when there is a conflict between two causal links.
   - A **conflict** exists when two causal links for some literal and its negation are not strictly ordered.

POP Example

Actions:

Action	PreCond	Effect
Go(*there*)	At(*here*)	At(*there*) ∧ ¬At(*here*)
Buy(*x*)	At(*store*) ∧ Sells(*store*, *x*)	Have(*x*)

Initial Plan:

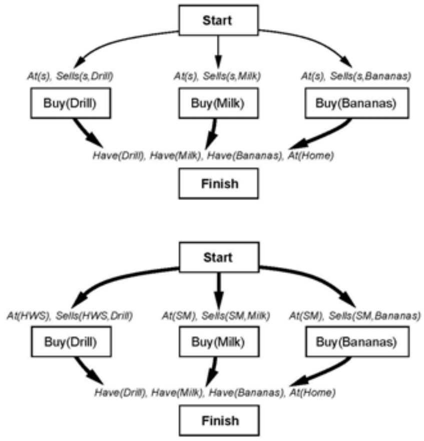
A Partial Plan I

Planners must commit to bindings for variables
Example: Goal: Have(Milk) Action: Buy(item,store)

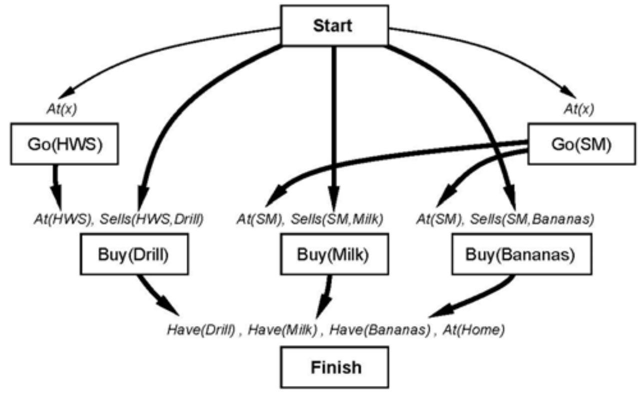
Principle of Least Commitment: Only make choices about things that you care about, leaving other details to be worked out later.
Buy(Milk,K-MART) versus Buy(Milk,store)

Fully instantiated plan: every variable is bound to a constant.

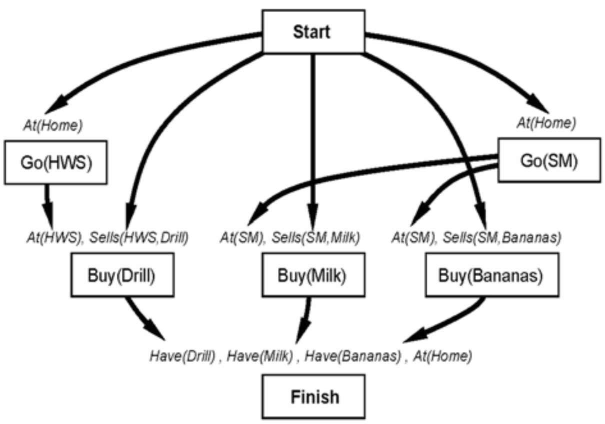
A Partial Plan II



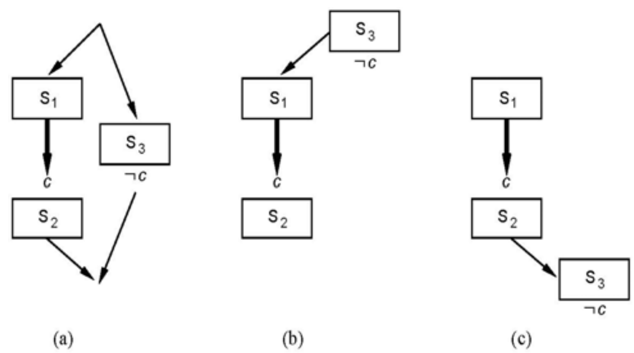
A Partial Plan III



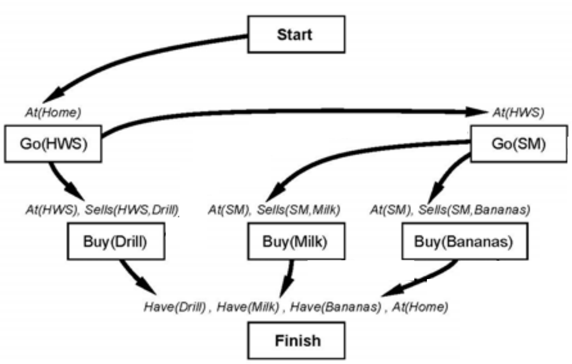
A Partial Plan IV



Protecting Causal Links



A Partial Plan IV’

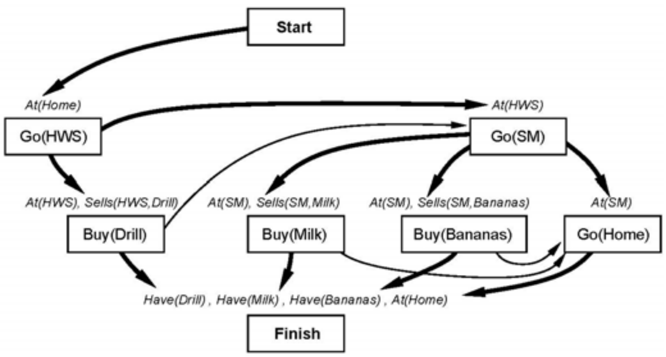


Achieving At(Home)

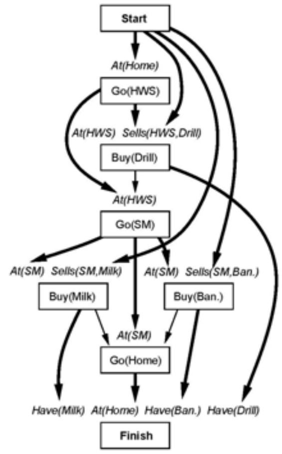
Candidate link	Threats
At(x) to initial state	Go(HWS), Go(SM)
At(x) to Go(HWS)	Go(SM)
At(x) to Go(SM)	At(SM) preconds of Buy(Milk), Buy(Bananas)

Solution: Link At(x) to Go(SM), but order Go(Home) to come after Buy(Bananas) and Buy(Milk).

A Partial Plan V



A Final Plan




```
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 --c--> S_need to LINKS(plan)
  add the ordering constraint S_add ≺ S_need to ORDERINGS(plan)
  if S_add is a newly added step from operators then
     add S_add to STEPS(plan)
     add Start ≺ S_add ≺ Finish to ORDERINGS(plan)

procedure RESOLVE-THREATS(plan)

  for each S_threat that threatens a link S_i --c--> S_j in LINKS(plan) do
     choose either
        Promotion: Add S_threat ≺ S_i to ORDERINGS(plan)
        Demotion: Add S_j ≺ S_threat to ORDERINGS(plan)
     if not CONSISTENT(plan) then fail
  end
```

Strengths of Partial-Order Planning Algorithms

- **Takes a huge state space problem and solves in only a few steps.**
- **Least commitment strategy means that search only occurs in places where sub-plans interact.**
- **Causal links allow planner to recognize when to abandon a doomed plan without wasting time exploring irrelevant parts of the plan.**

Practical Planners

STRIPS approach is insufficient for many practical planning problems. Can't express:

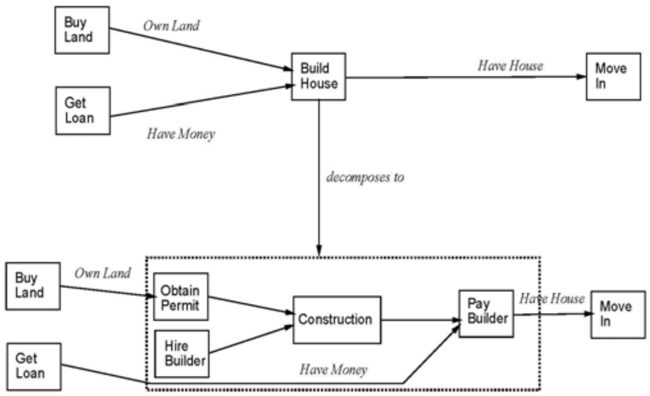
- **Resources:** Operators should incorporate resource consumption and generation. Planners have to handle constraints on resources efficiently.
- **Time:** Real-world planners need a better model of time.
- **Hierarchical plans:** need the ability to specify plans at varying levels of details.

Also need to incorporate heuristics for guiding search.

Planning Graphs

- Data structure (graphs) that represent plans, and can be efficiently constructed, and that allows for better heuristic estimates.
- **Graphplan:** algorithm that processes the planning graph, using backward search, to extract a plan.
- **SATPlan:** algorithm that translates a planning problem into propositional axioms and applies a CSP algorithm to find a valid plan.
- **Take CS672 / CS475 to learn more!!**

Hierarchical Planning



Spacecraft Assembly, Integration and Verification (AIV)

- **OPTIMUM-AIV used by the European Space Agency to AIV spacecraft.**
- **Generates plans and monitors their execution – ability to re-plan is the principle objective.**
- **Uses O-Plan architecture – like partial-order planner, but can represent time, resources and hierarchical plans. Accepts heuristics for guiding search and records its reasons for each choice.**

Scheduling for Space Missions

- **Planners have been used by ground teams for the Hubble space telescope and for the Voyager, UOSAT-II and ERS-1.**
- **Goal: coordinate the observational equipment, signal transmitters and altitude and velocity-control mechanism in order to maximize the value of the information gained from observations while obeying resource constraints on time and energy.**