
Planning

Philipp Koehn

10 March 2026



Outline



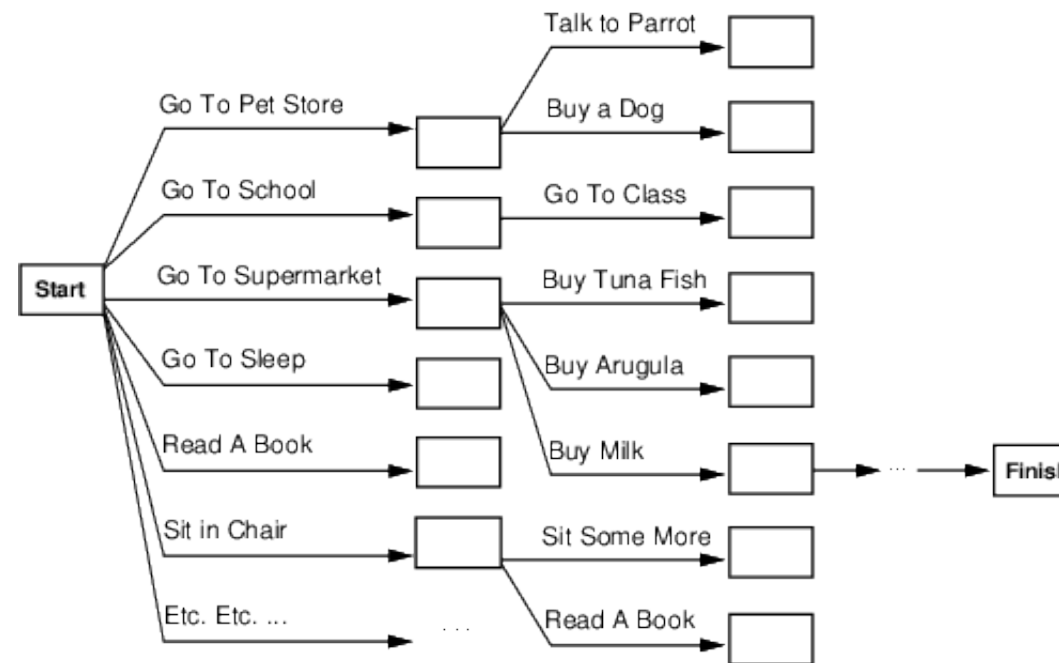
1

- Search vs. planning
- STRIPS operators
- Partial-order planning
- The real world
- Conditional planning
- Monitoring and replanning

search vs. planning

Search vs. Planning

- Consider the task *get milk, bananas, and a cordless drill*
- Standard search algorithms seem to fail miserably



- Too many choices, no immediate feedback

Search vs. Planning

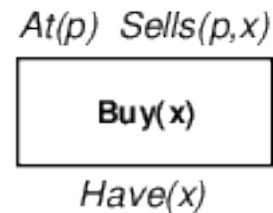


- Planning systems do the following
 1. improve action and goal representation to allow selection
 2. divide-and-conquer by **subgoal**ing
 3. relax requirement for sequential construction of solutions
- Differences

	Search	Planning
States	Data structures	Logical sentences
Actions	Program code	Preconditions/outcomes
Goal	Program code	Logical sentence (conjunction)
Plan	Sequence from S_0	Constraints on actions

partial-order planning

STRIPS Operators



ACTION: *Buy(x)*
PRECONDITION: *At(p), Sells(p, x)*
EFFECT: *Have(x)*

- Note: this abstracts away many important details!
- Restricted language \implies efficient algorithm
 - Precondition: conjunction of positive literals
 - Effect: conjunction of literals

Partially Ordered Plans



- **Partially ordered** collection of steps with
 - **Start step** has the initial state description as its effect
 - **Finish step** has the goal description as its precondition
 - **causal links** from outcome of one step to precondition of another
 - **temporal ordering** between pairs of steps■
- A plan is **complete** iff every precondition is achieved
- A precondition is **achieved** iff it is the effect of an earlier step and no **possibly intervening** step undoes it

Example



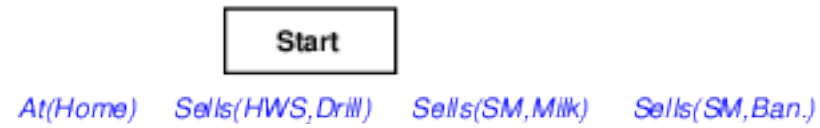
Start

At(Home) Sells(HWS,Drill) Sells(SM,Milk) Sells(SM,Ban.)

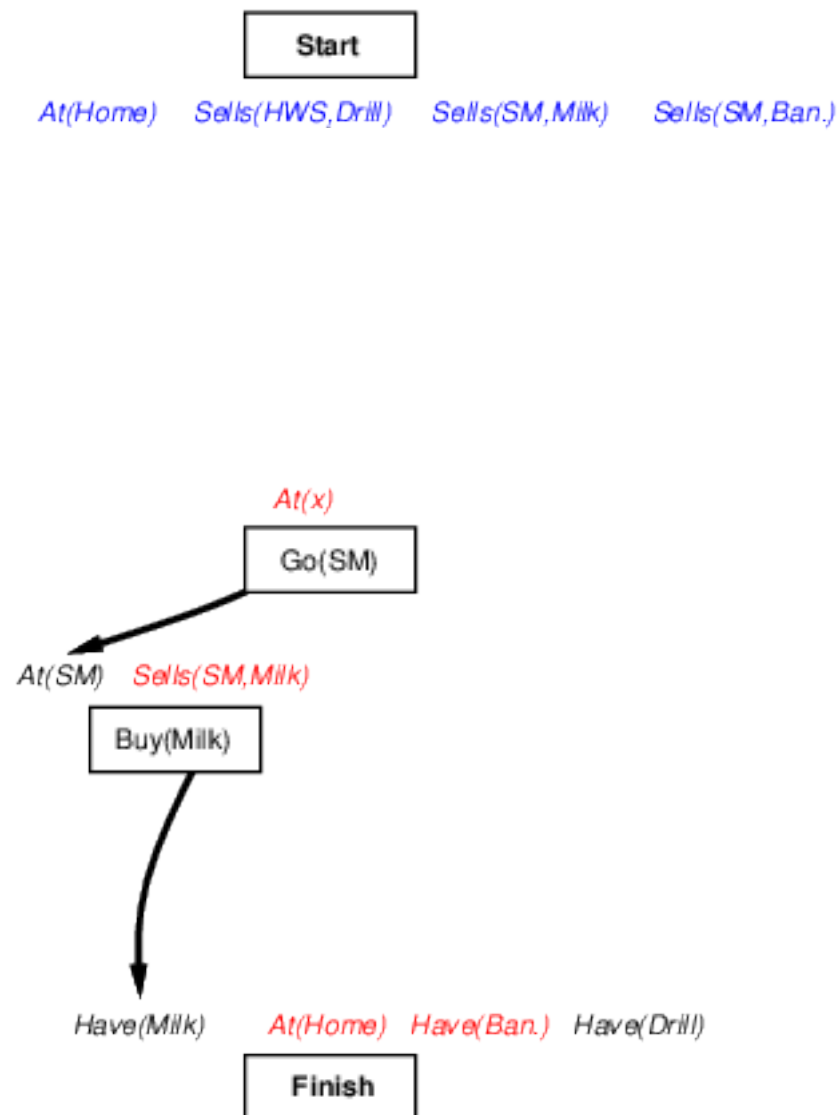
Have(Milk) At(Home) Have(Ban.) Have(Drill)

Finish

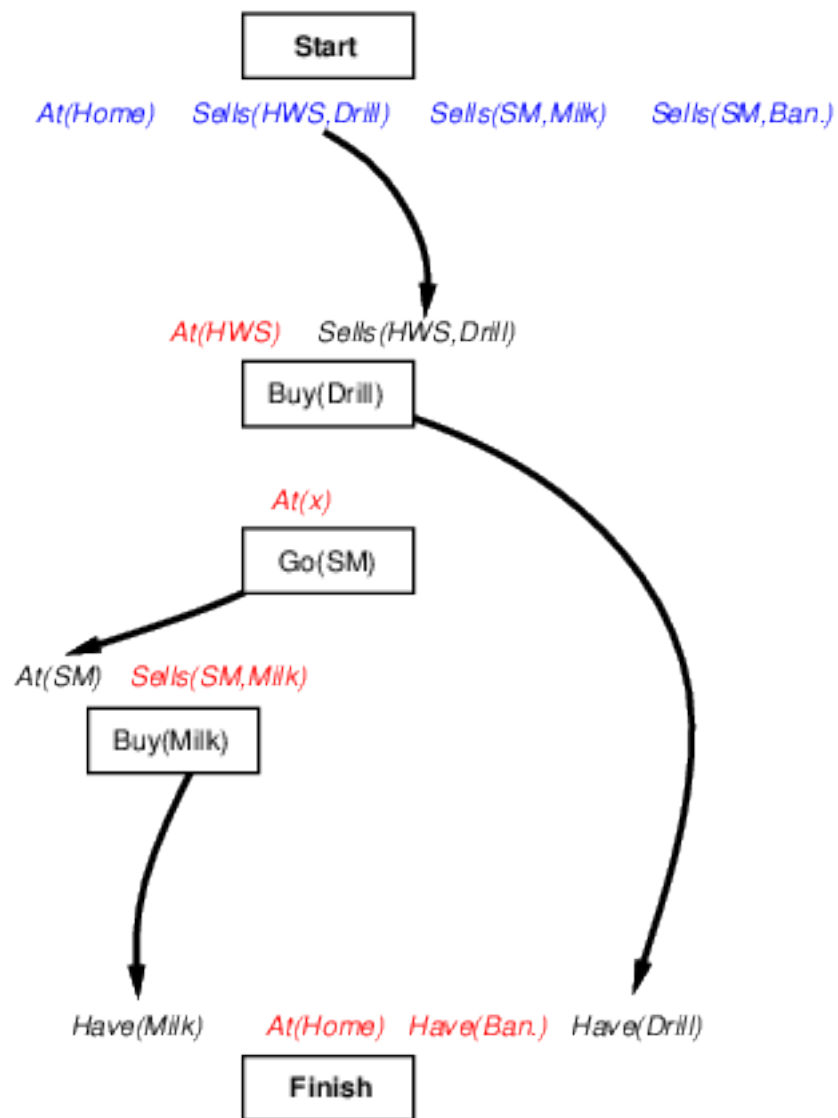
Example



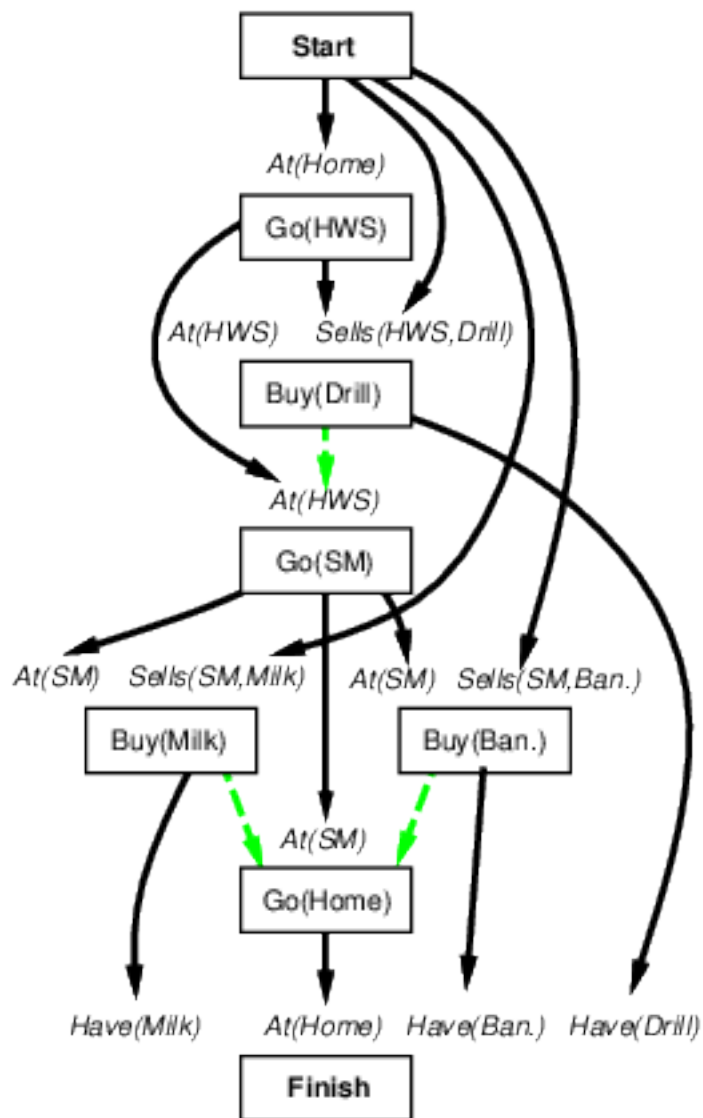
Example



Example



Example



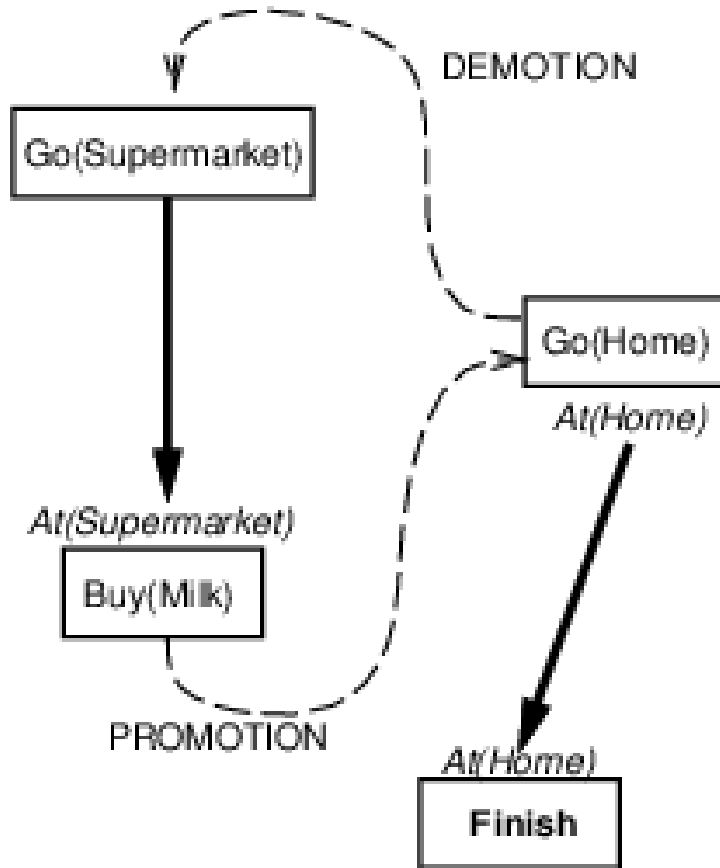
Planning Process



- Operators on partial plans
 - **add a link** from an existing action to an open condition
 - **add a step** to fulfill an open condition
 - **order** one step wrt another to remove possible conflicts
- Gradually move from incomplete/vague plans to complete, correct plans
- Backtrack if an open condition is unachievable or if a conflict is unresolvable

Clobbering and Promotion/Demotion

- A **clobberer** is a potentially intervening step that destroys the condition achieved by a causal link. E.g., $Go(Home)$ clobbers $At(Supermarket)$:



Demotion: put before $Go(Supermarket)$

Promotion: put after $Buy(Milk)$

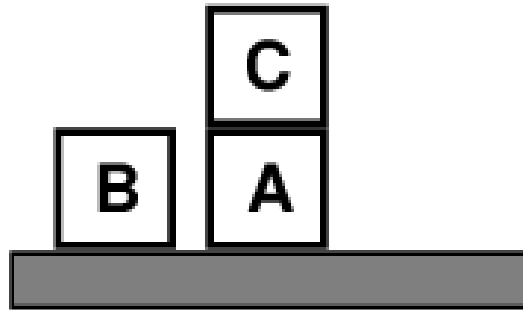
Properties of Partially Ordered Plans



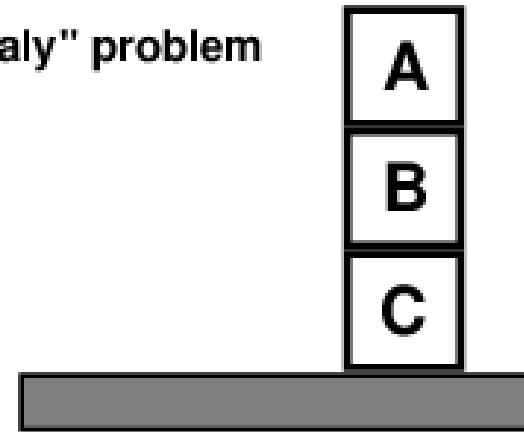
- Nondeterministic algorithm: backtracks at **choice** points on failure
 - choice of S_{add} to achieve S_{need}
 - choice of demotion or promotion for clobberer
 - selection of S_{need} is irrevocable■
- Partially Ordered Plans is sound, complete, and **systematic** (no repetition)
- Extensions for disjunction, universals, negation, conditionals
- Can be made efficient with good heuristics derived from problem description
- Particularly good for problems with many loosely related subgoals

Example: Blocks World

"Sussman anomaly" problem

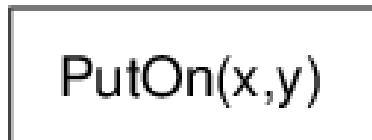


Start State



Goal State

$Clear(x) \ On(x,z) \ Clear(y)$



$\sim On(x,z) \ \sim Clear(y)$
 $Clear(z) \ On(x,y)$

$Clear(x) \ On(x,z)$



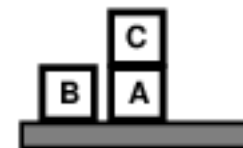
$\sim On(x,z) \ Clear(z) \ On(x, Table)$

+ several inequality constraints

Example

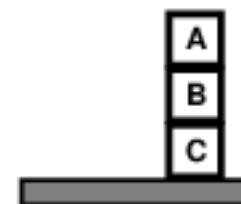
START

On(C,A) On(A,Table) Cl(B) On(B,Table) Cl(C)

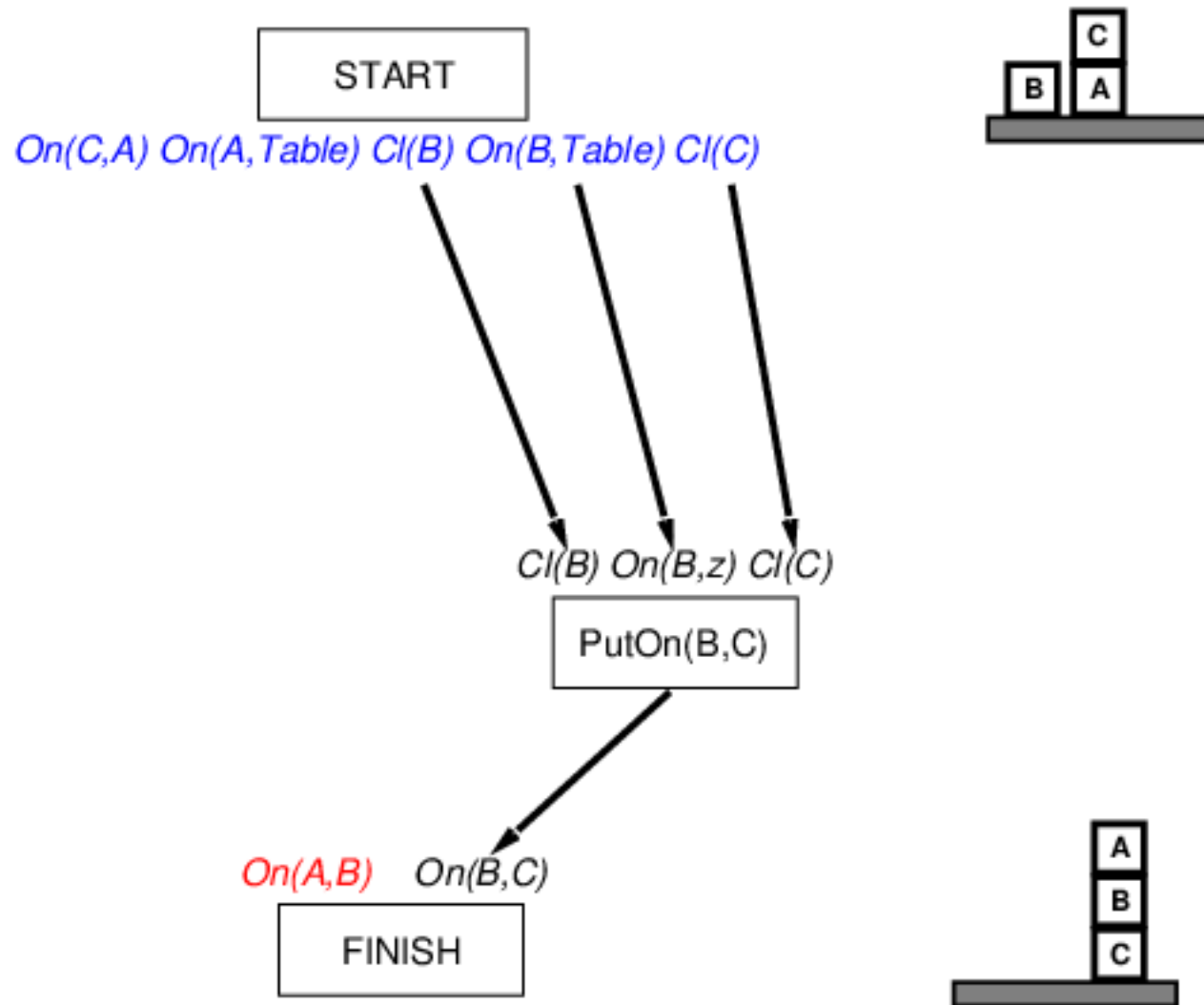


On(A,B) On(B,C)

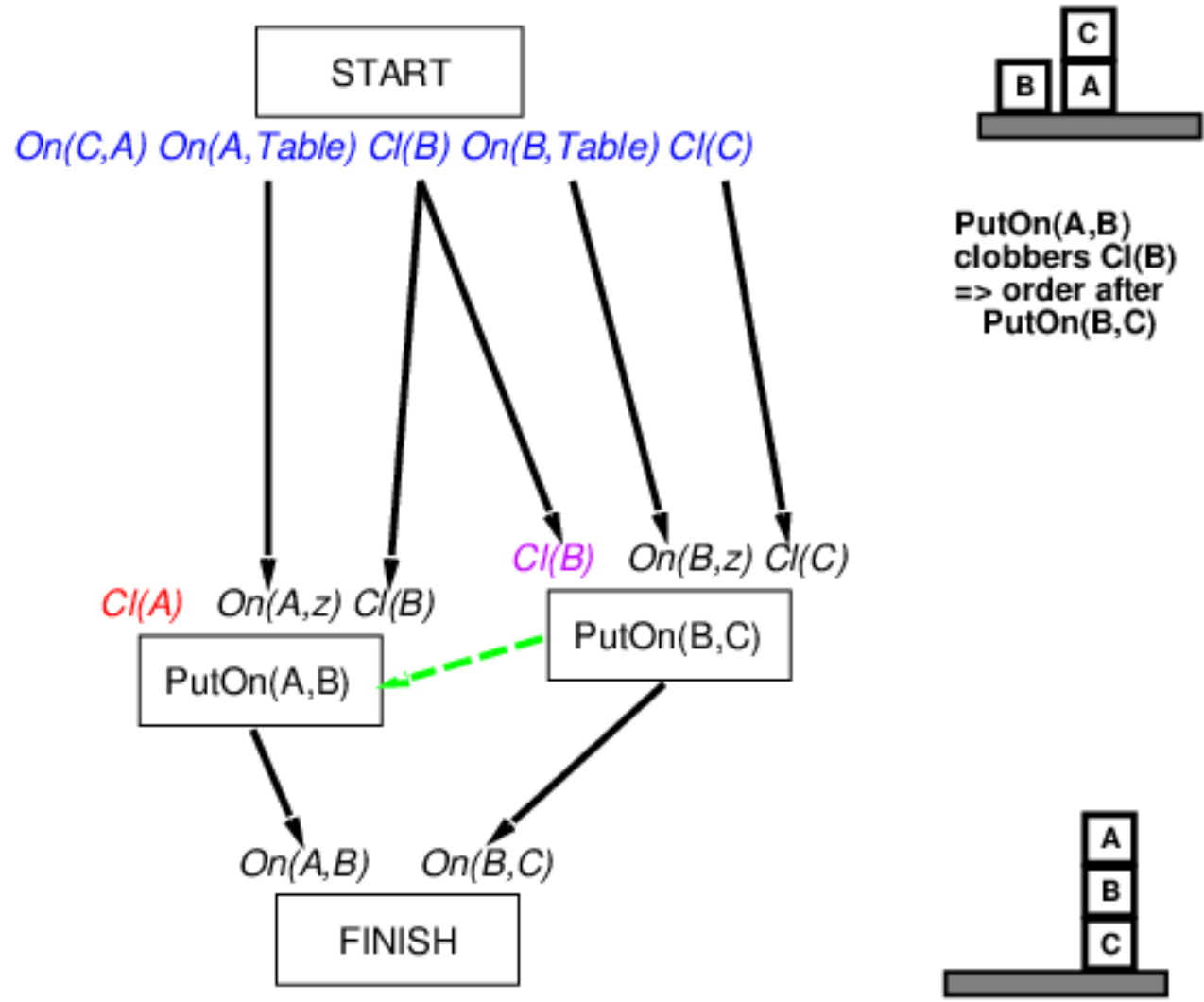
FINISH



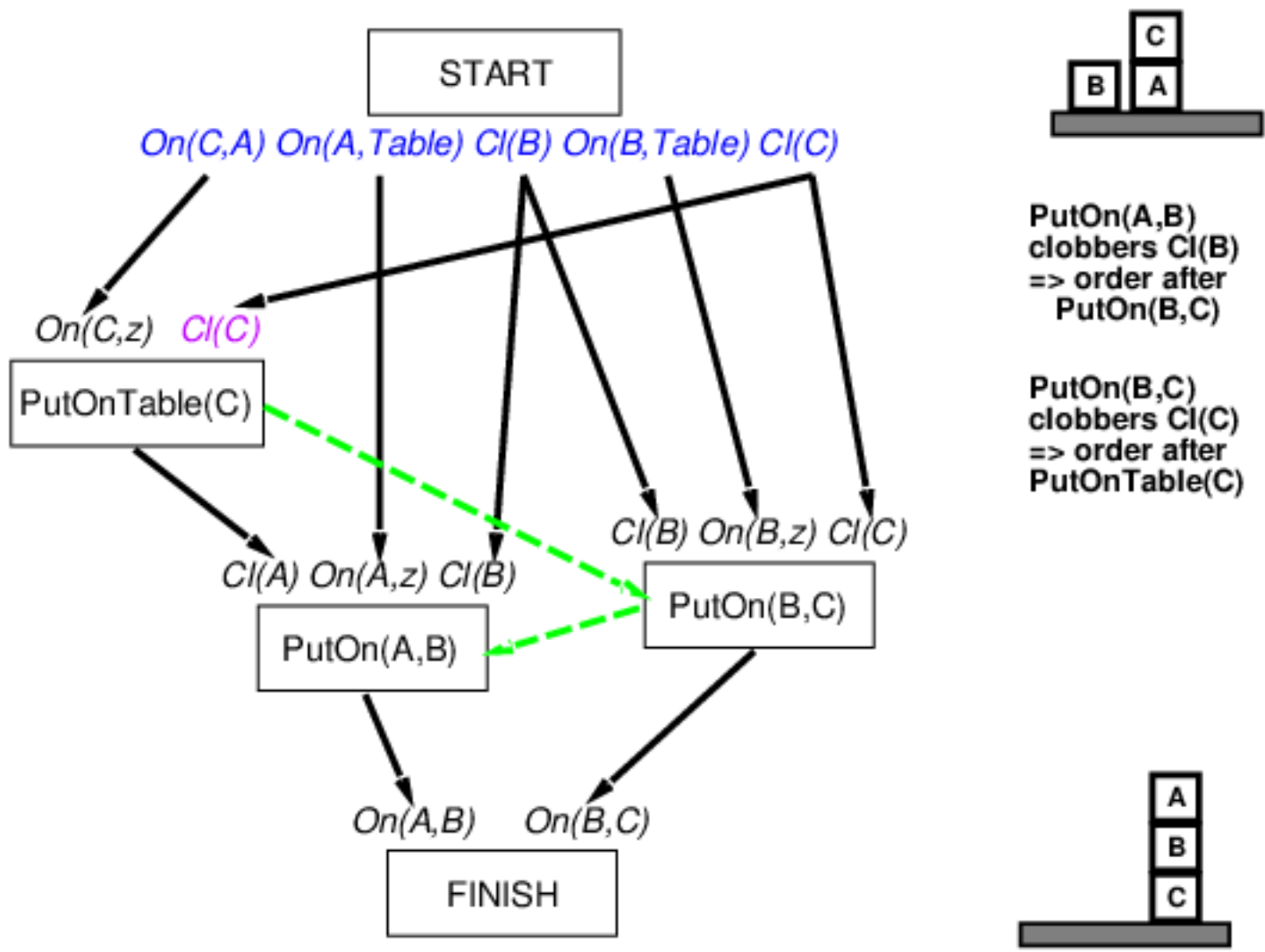
Example



Example



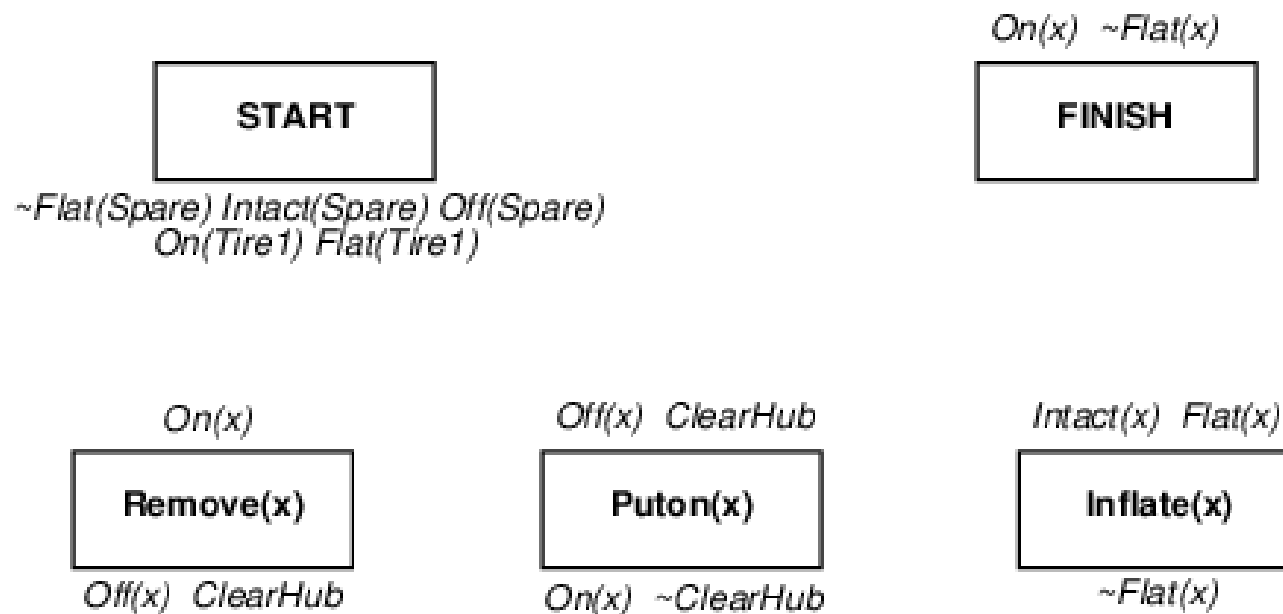
Example





the real world

The Real World



Things Go Wrong

- **Incomplete information**

- unknown preconditions, e.g., $Intact(Spare)$?
- disjunctive effects, e.g., $Inflate(x)$ causes $Inflated(x) \vee SlowHiss(x) \vee Burst(x) \vee BrokenPump \vee \dots$ ■

- **Incorrect information**

- current state incorrect, e.g., spare NOT intact
- missing/incorrect postconditions in operators ■

- **Qualification problem** can never finish listing all

- required preconditions of actions
- possible conditional outcomes of actions

- **Conformant** or **sensorless planning**

Devise a plan that works regardless of state or outcome

Such plans may not exist

- **Conditional planning**

Plan to obtain information (**observation actions**)

Subplan for each contingency, e.g.,

[*Check(Tire1)*, **if** *Intact(Tire1)* **then** *Inflate(Tire1)* **else** *CallAAA*]

Expensive because it plans for many unlikely cases

- **Monitoring/Replanning**

Assume normal states, outcomes

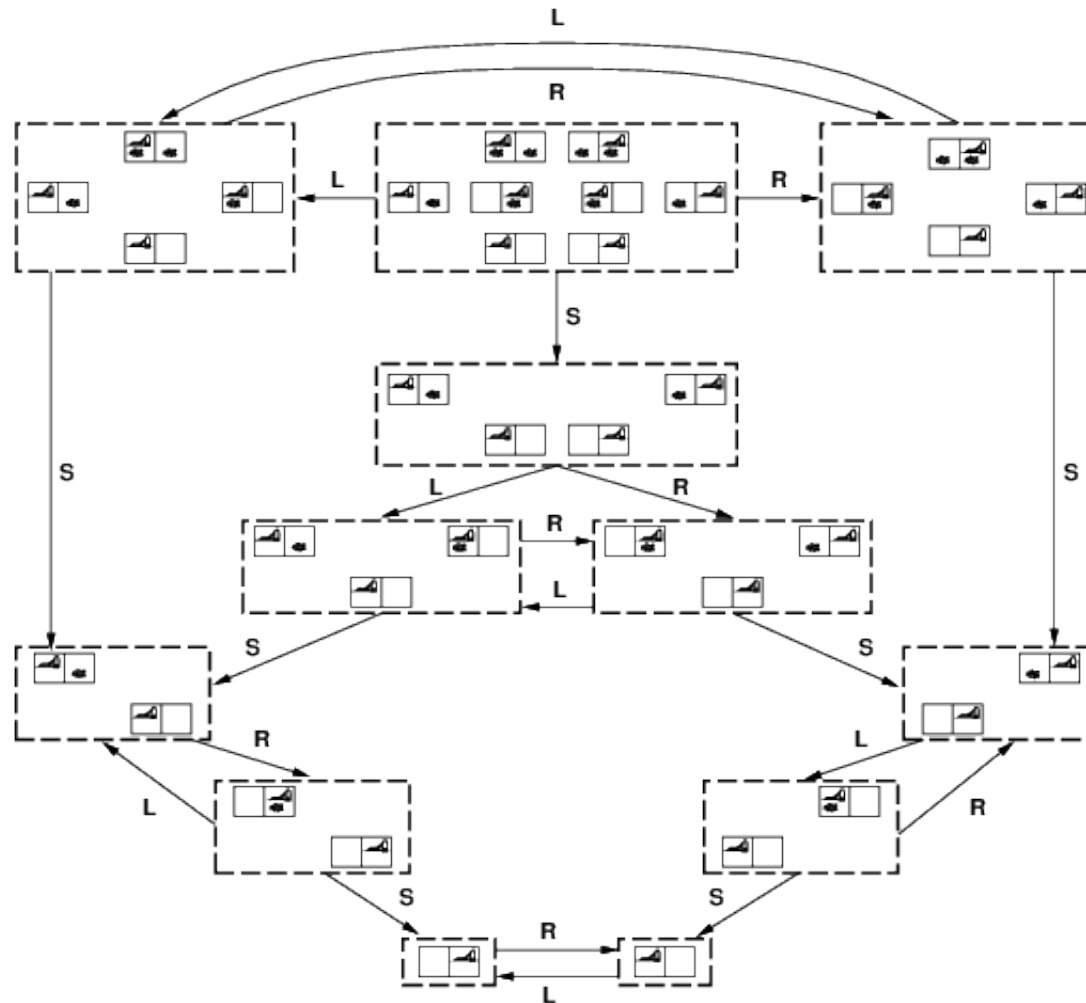
Check progress *during execution*, replan if necessary

Unanticipated outcomes may lead to failure (e.g., no AAA card)

⇒ Really need a combination; plan for likely/serious eventualities, deal with others when they arise, as they must eventually.

Conformant Planning

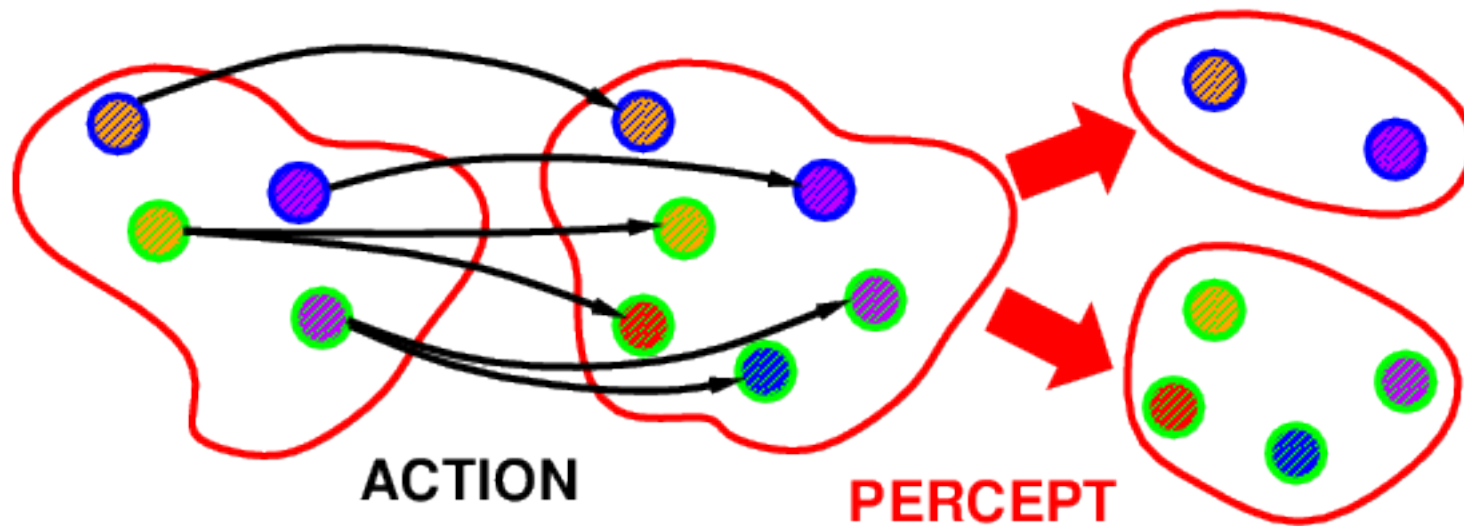
- Search in space of **belief states** (sets of possible actual states)



conditional planning

Conditional Planning

- If the world is nondeterministic or partially observable then percepts usually *provide information*, i.e., *split up* the belief state

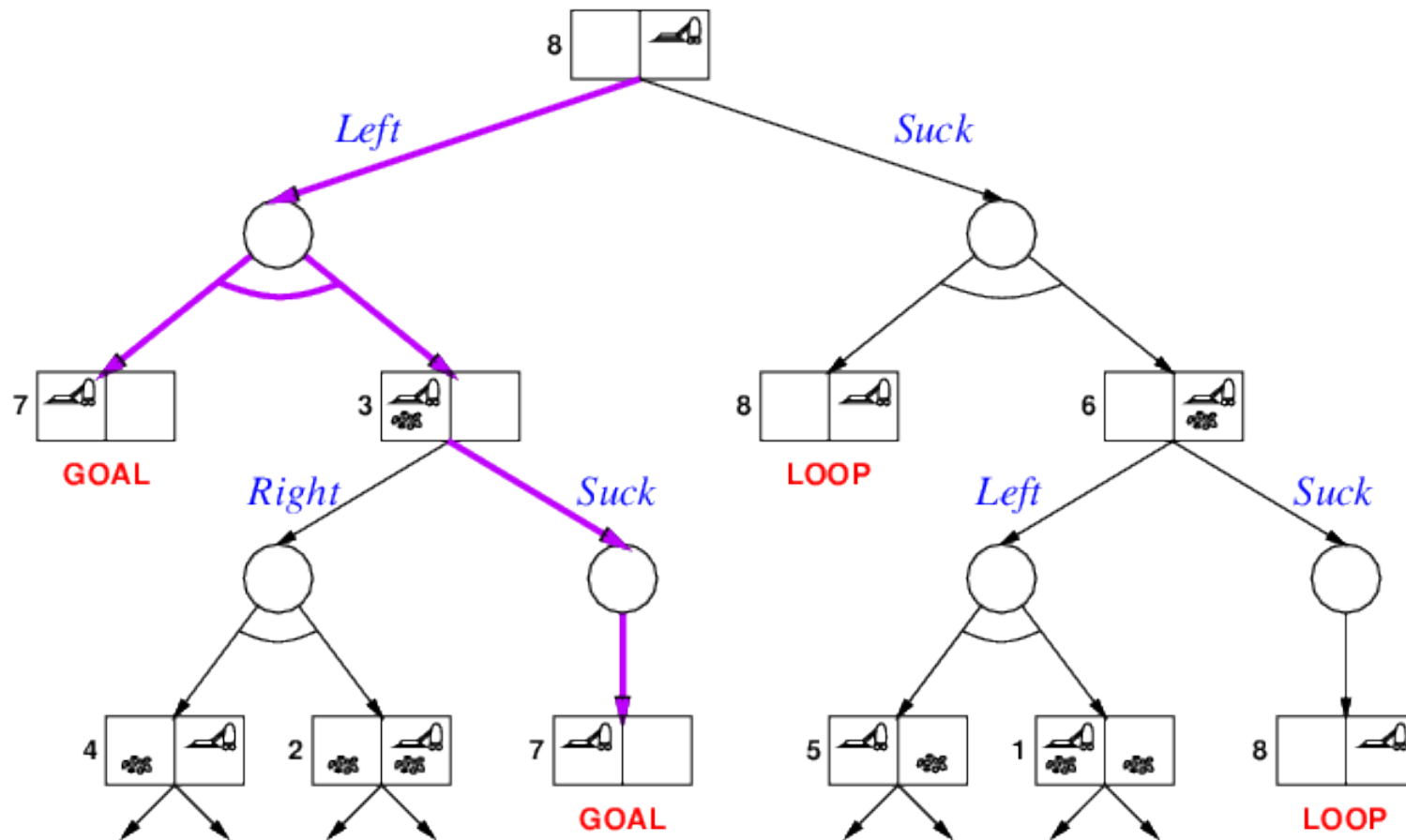


Conditional Planning

- Conditional plans check (any consequence of KB +) percept
- [..., **if** C **then** $Plan_A$ **else** $Plan_B$, ...]
- Execution: check C against current KB, execute “then” or “else” ■
- Need *some* plan for *every* possible percept
 - game playing: *some* response for *every* opponent move
 - backward chaining: *some* rule such that *every* premise satisfied
- AND–OR tree search (very similar to backward chaining algorithm)

Example

- Sucking or arriving may dirty a clean square



monitoring and replanning

Execution Monitoring

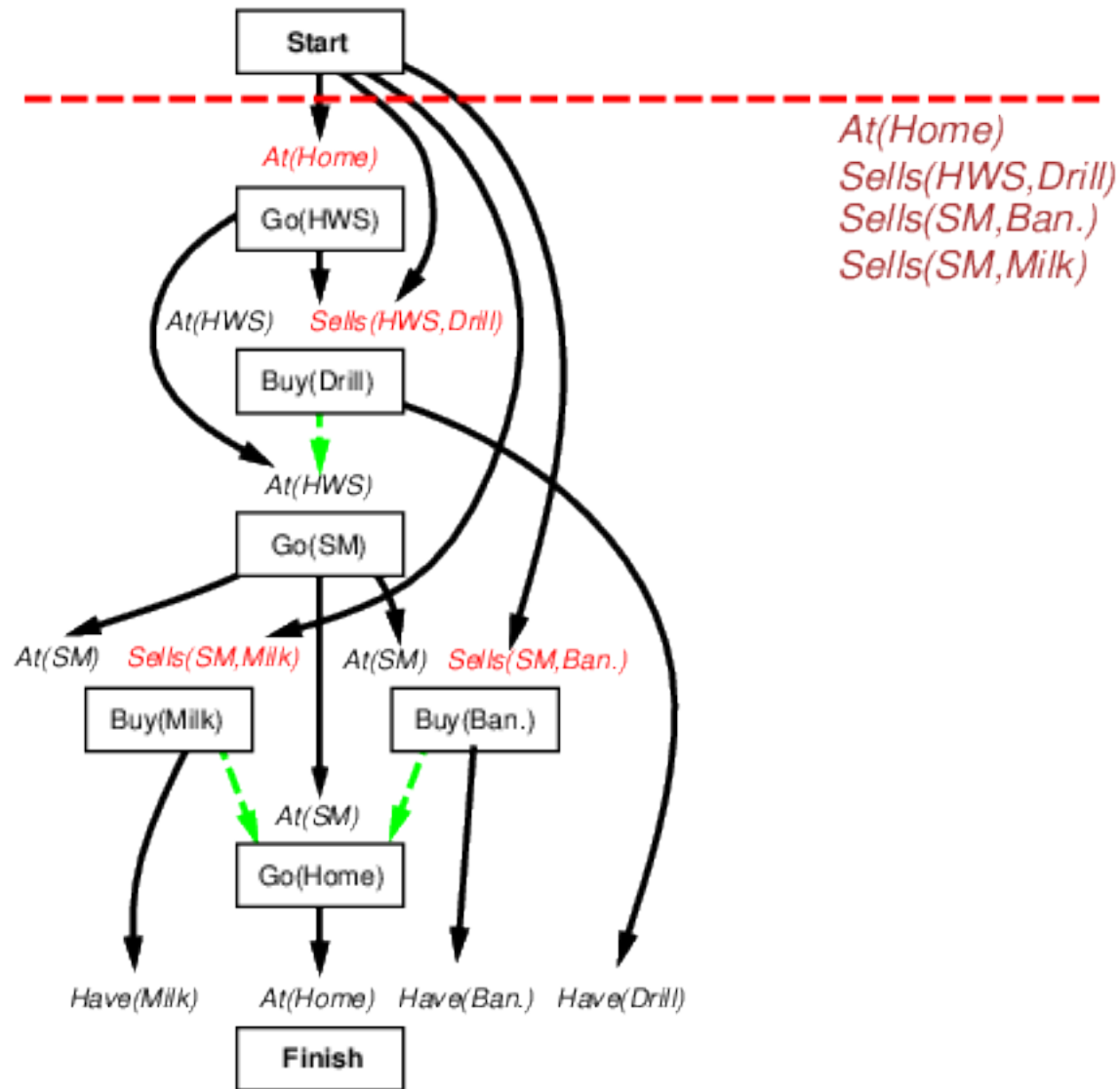
- Plan with Partially Ordered Plans algorithms
- Process plan, one step at a time
- Validate planned conditions against perceived reality■
- “Failure” = preconditions of *remaining plan* not met
- Preconditions of remaining plan
 - = all preconditions of remaining steps not achieved by remaining steps
 - = all causal links *crossing* current time point

Responding to Failure

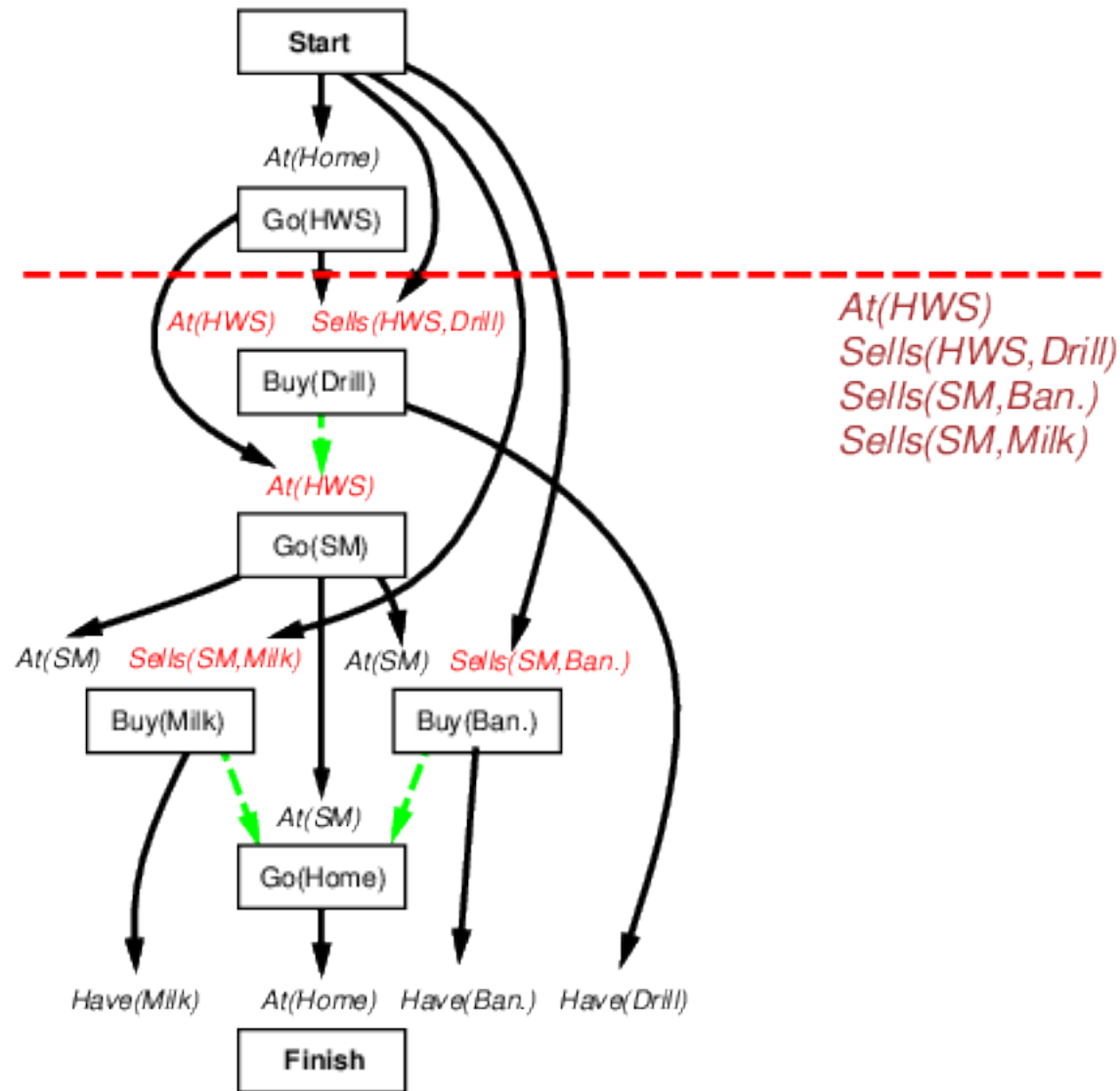


- Run Partially Ordered Plans algorithms again
- Resume Partially Ordered Plans to achieve open conditions from current state
- **Integrated Planning, Execution, and Monitoring**
 - keep updating *Start* to match current state
 - links from actions replaced by links from *Start* when done

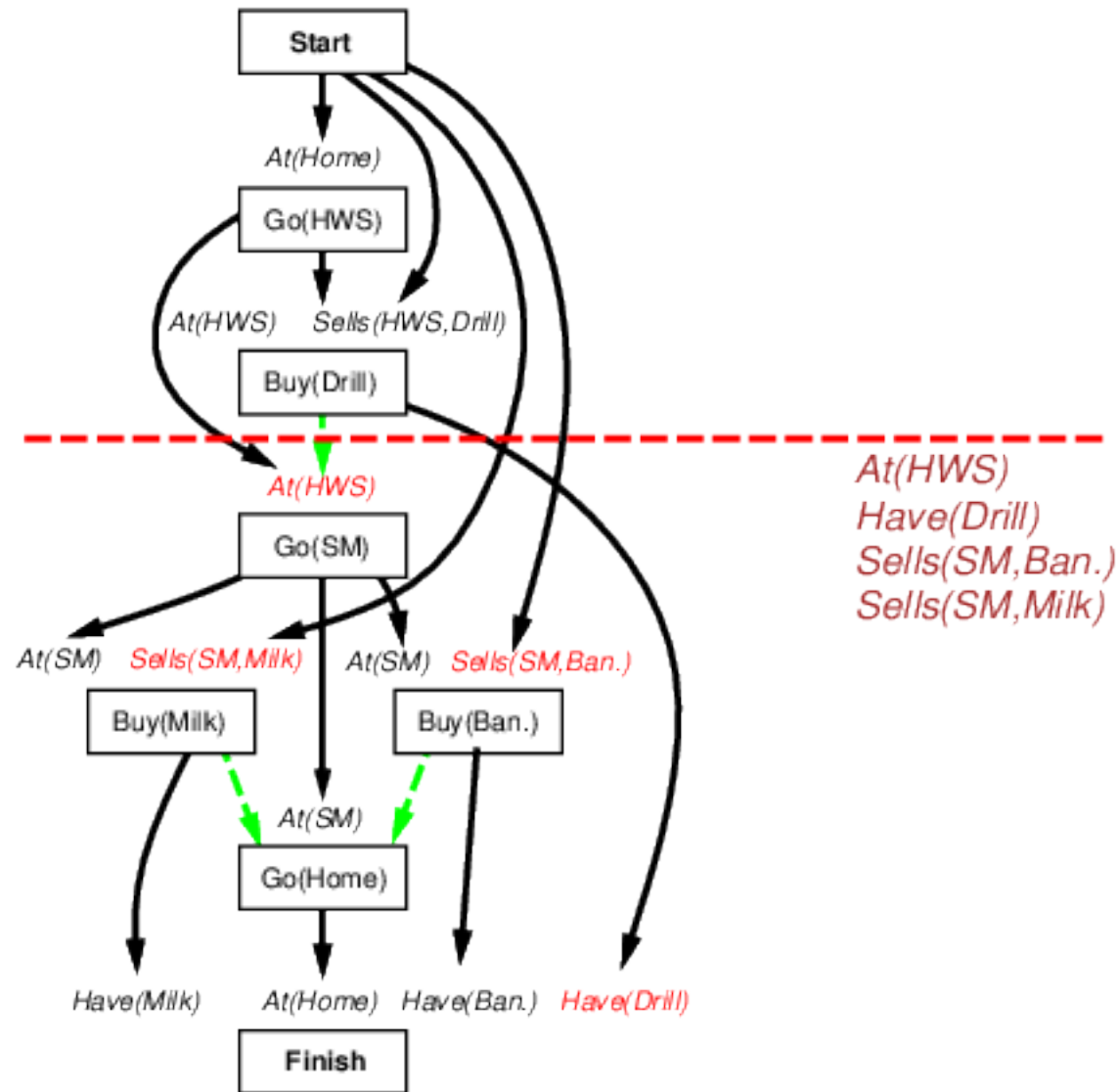
Example



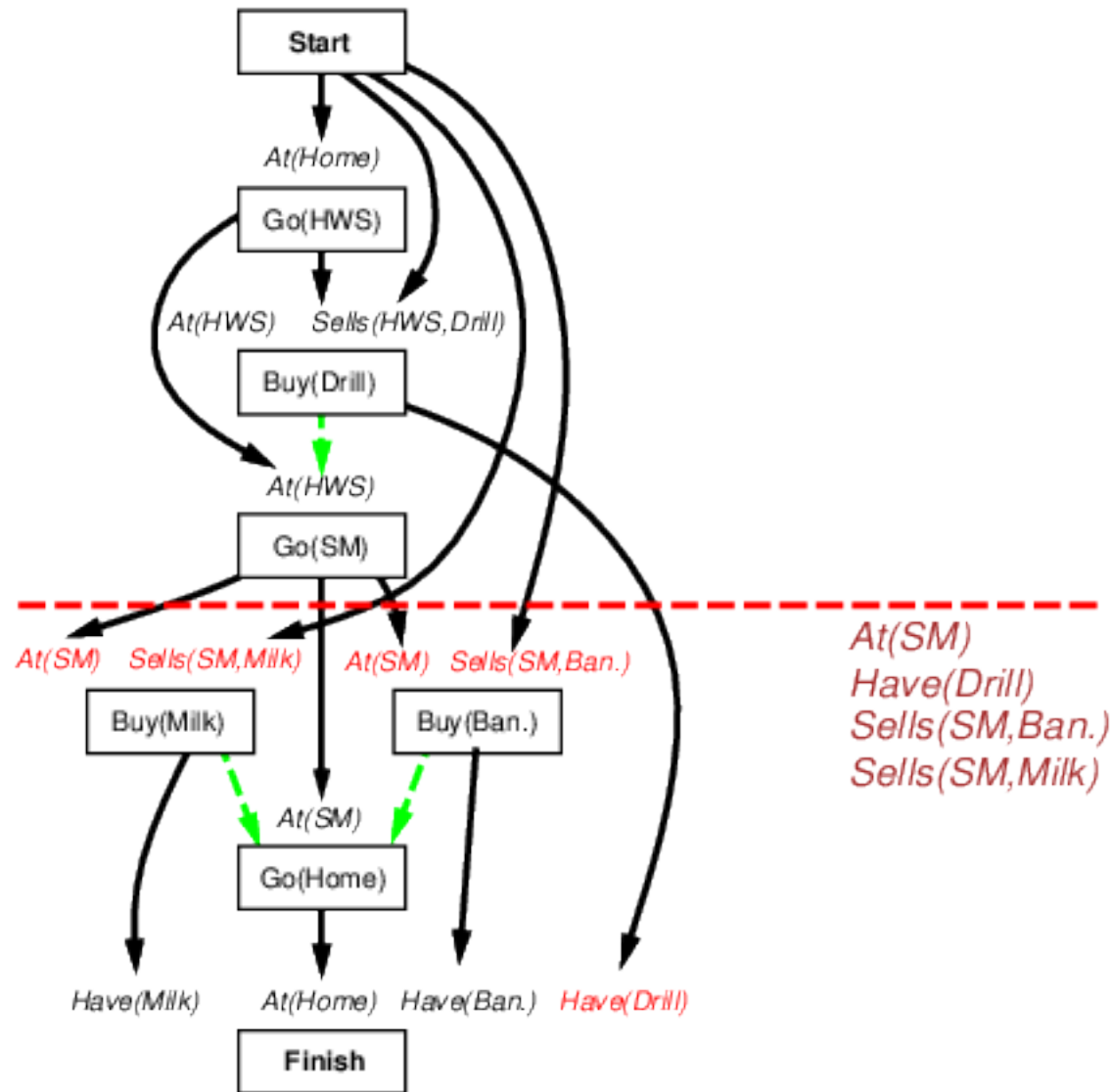
Example



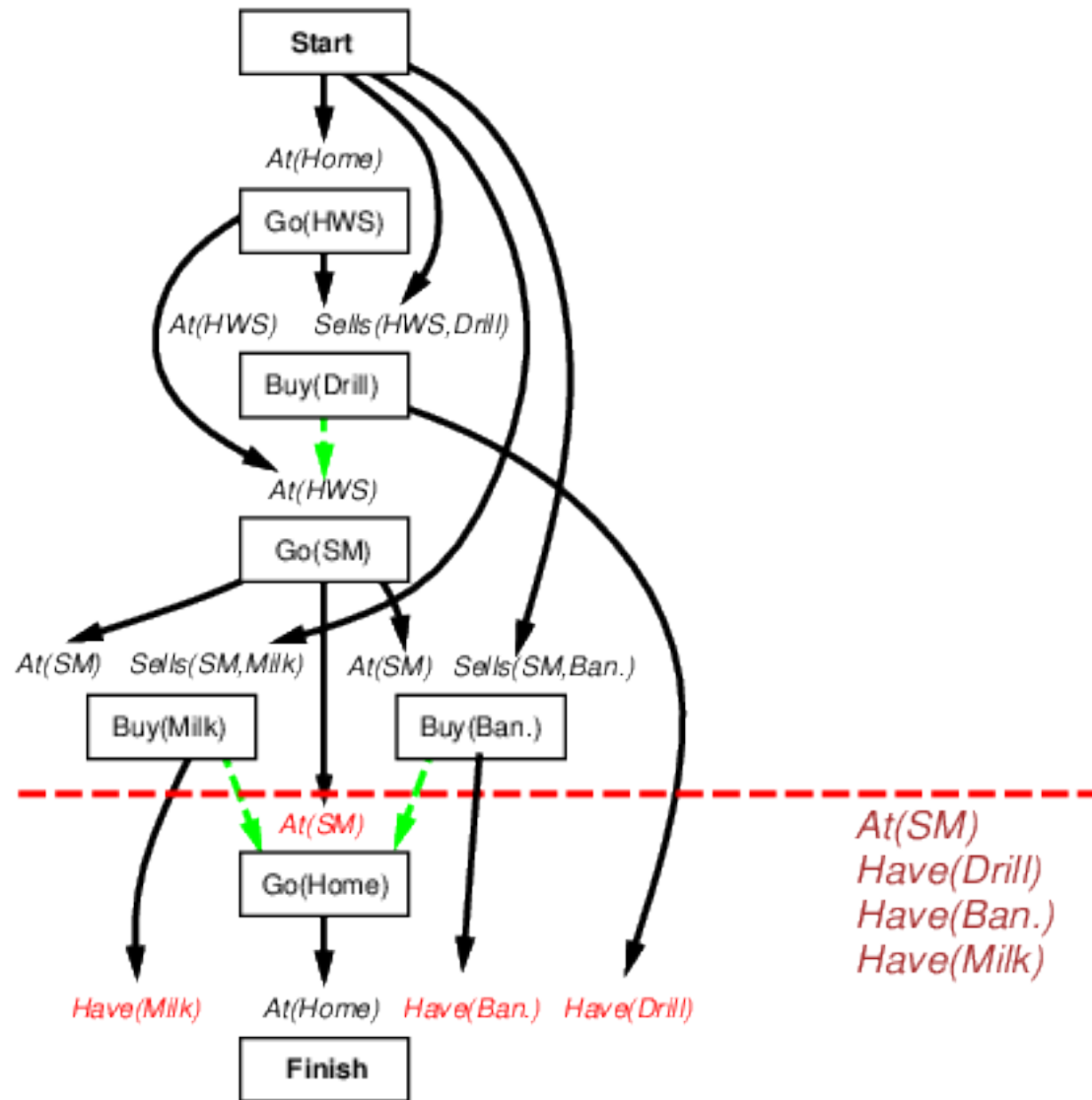
Example



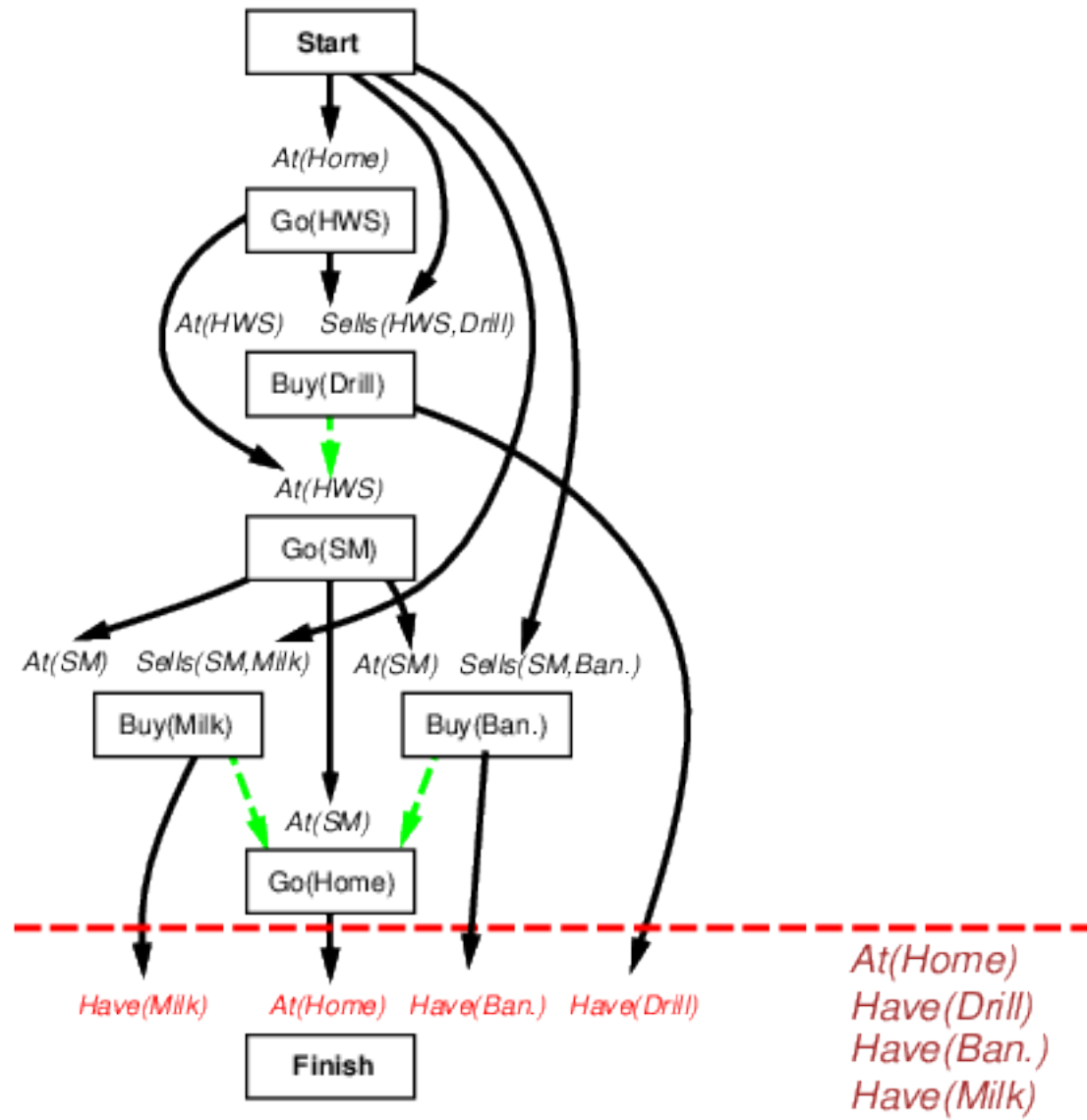
Example



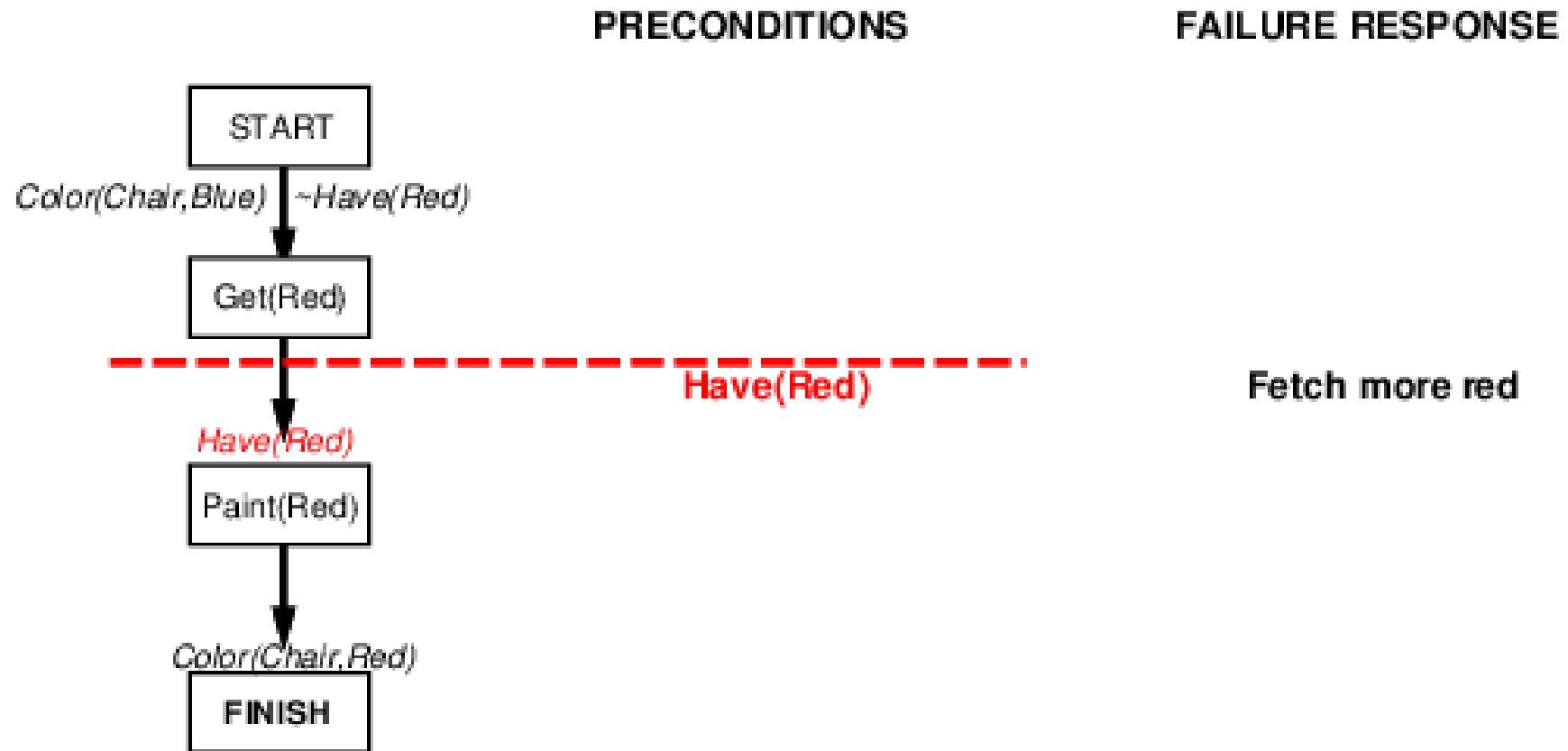
Example



Example



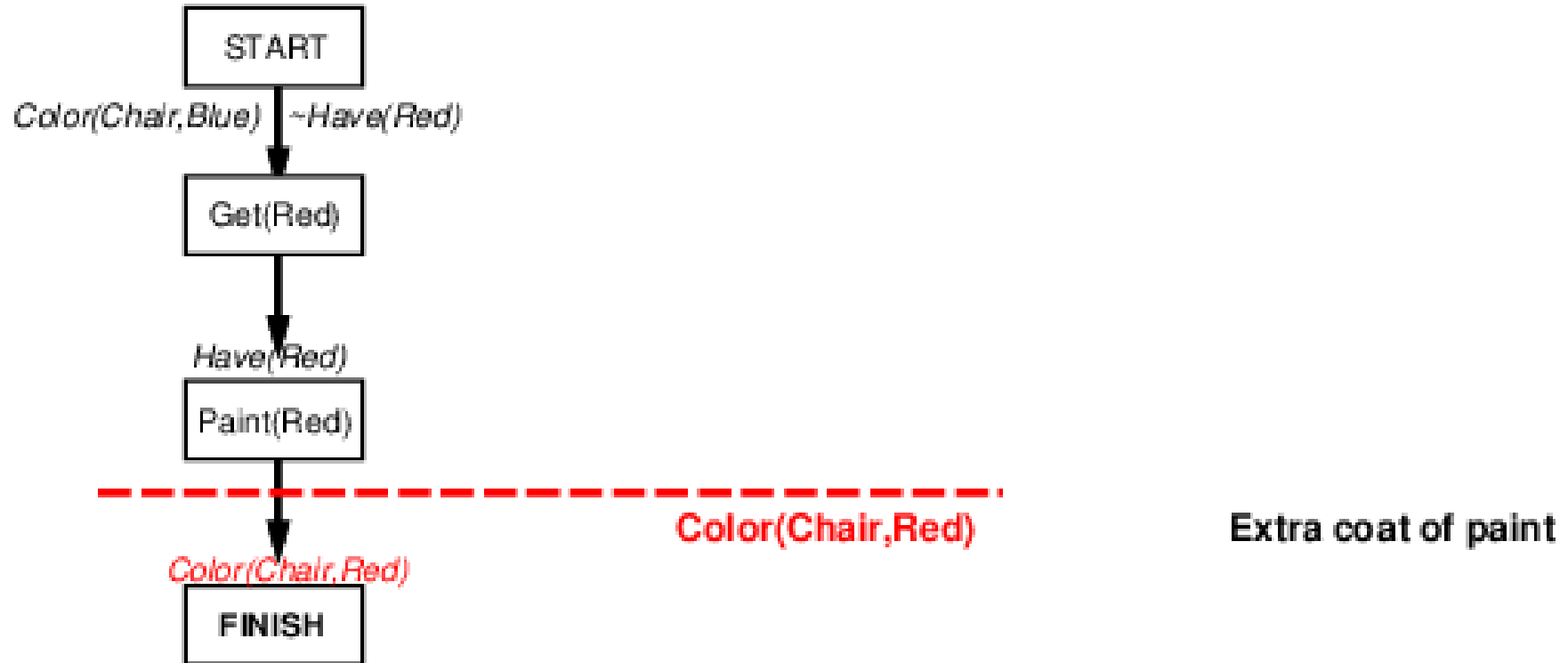
Emergent Behavior



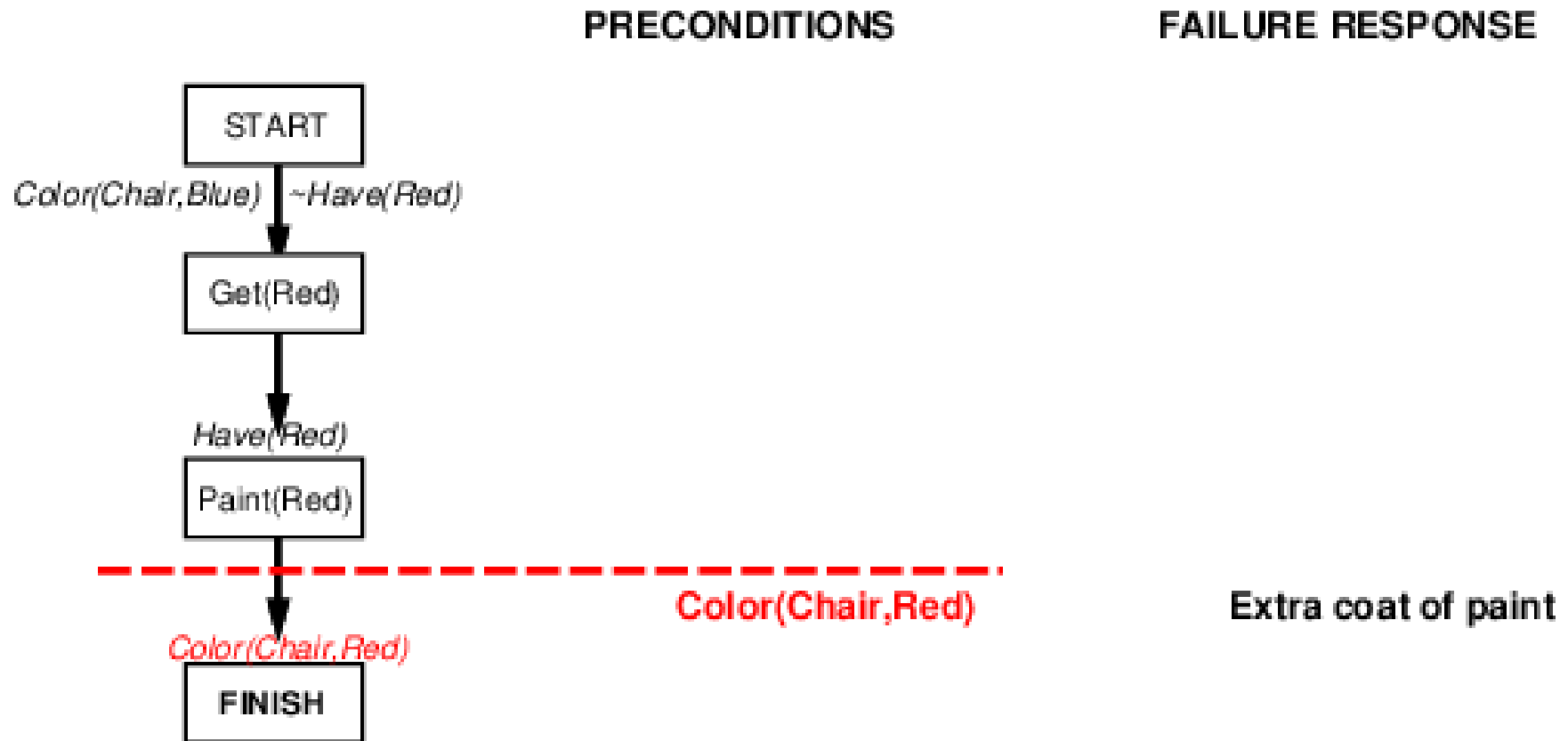
Emergent Behavior

PRECONDITIONS

FAILURE RESPONSE



Emergent Behavior



- “Loop until success” behavior *emerges* from interaction between monitor/replan agent design and uncooperative environment

Summary



- Planning
 - break down problem into subgoals
 - search for plans for subgoals
 - merge sub-plans
 - Defined actions in terms of preconditions and effects
 - Partially Ordered Plans algorithm
 - Clobbering: need to deal with steps that destroy clausal link in plan
 - Real world: incomplete and incorrect information
- ⇒ conformant or conditional planning, monitoring and replanning