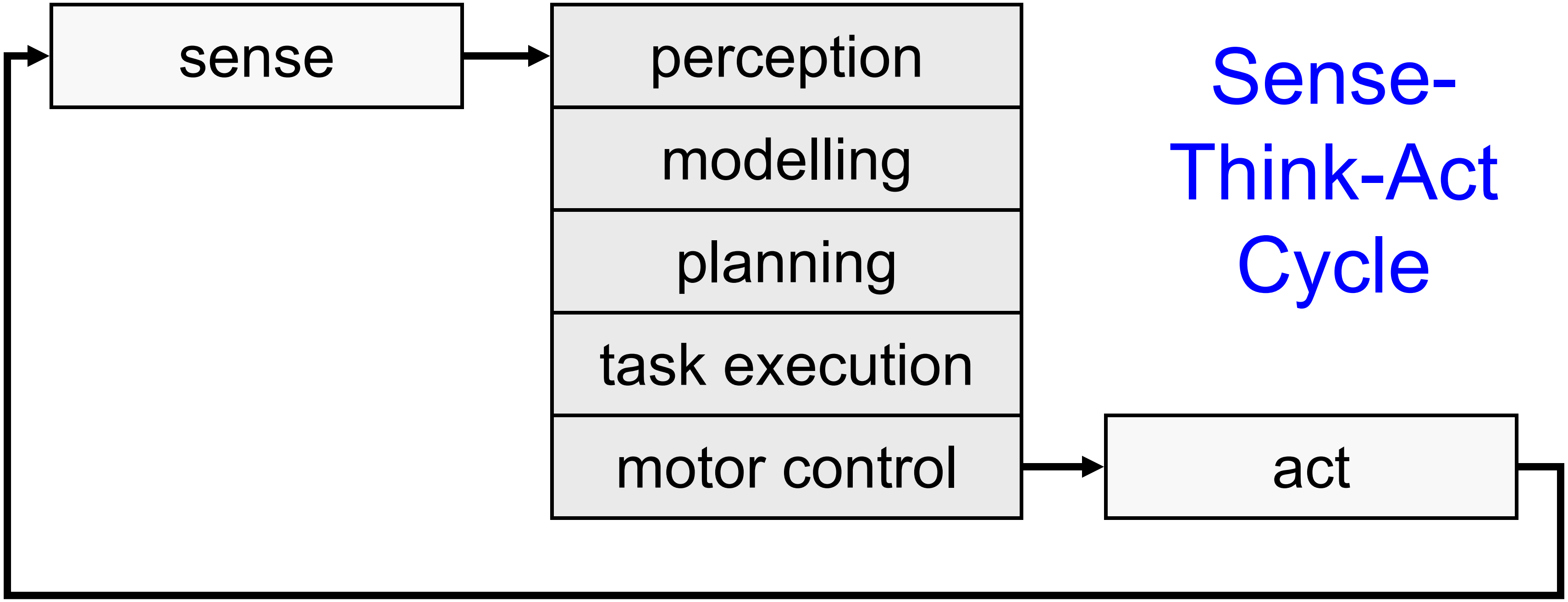
Planning

CPSC 470 – Artificial Intelligence
Brian Scassellati

A Simple Planning Agent



sense

perception

modelling

planning

task execution

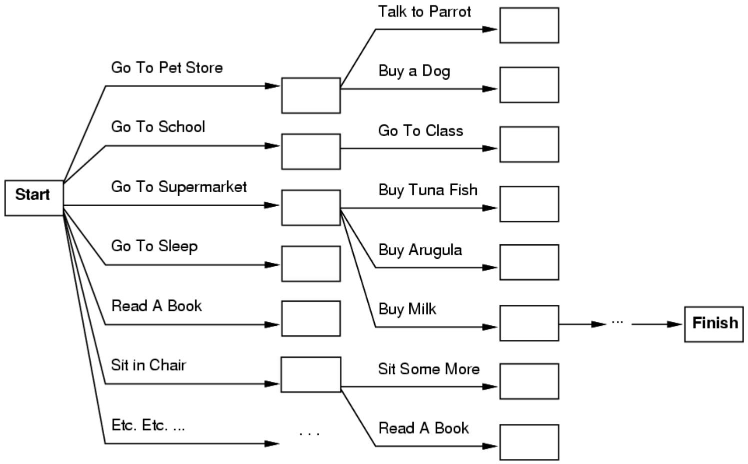
motor control

act

Sense-
Think-Act
Cycle

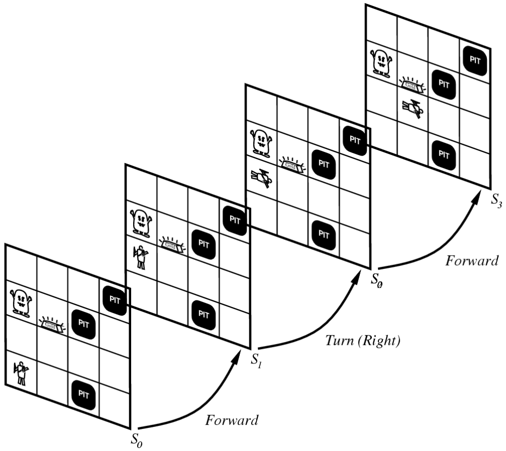
- Minimal interaction with environment
- Batch processing of the plan before any actions are taken

Planning: Haven't we done this already?



- Planning is like search…
  - Consider all the possible actions you could take and build a search tree
  - Rather than labeling individual situations, we describe general operators that change parts of the world state

Planning:
Haven't we done this already?



- Situations are indexed

  $At(Agent,[1,1], S_0) \wedge At(Agent,[1\ 2], S_1)$
- Changes from one situation to the next

  $Result(Forward, S_0) \Rightarrow S_1$

  $Result(Turn(R), S_1) \Rightarrow S_2$

- Planning is like inference…
  - Describe actions as operators in a logic, then just prove that the solution that you want exists
  - Need to have some efficiency considerations in dealing with large numbers of actions on complex environments

Planning: Haven’t we done this already?

- Yes, we can frame the planning problem in different ways
  - But there are some useful things about planning that make both search and inference inefficient
  - It is such a common problem that we study it individually

Key Ideas to Planning

1. “Open Up” the representation of states, goals, and actions
   - Only pick actions that help you achieve the goal… don’t pick at random
2. Planner can add actions wherever they are needed
   - No need to keep to an incremental sequence… can grow the “chain of action” at any point
3. Most parts of the world are independent
   - Divide-and-conquer becomes feasible

STRIPS planner

STanford Research Institute Problem Solver

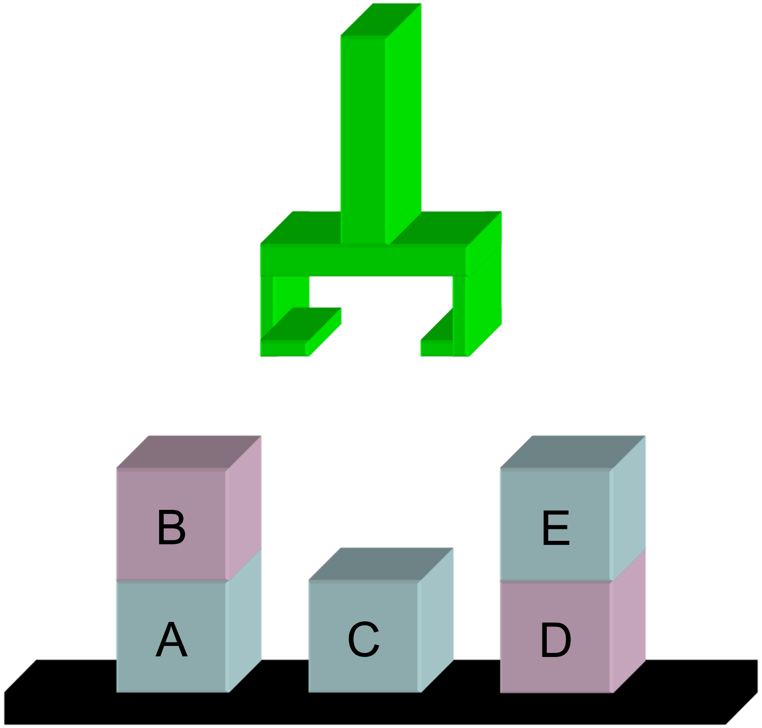
- “Holy Roman Empire” naming
- Represent states and goals in first-order logic

  $At(Home) \wedge Have(Milk) \wedge Have(Drill) \wedge Have(Banana)$
- Assume existential quantification of variables

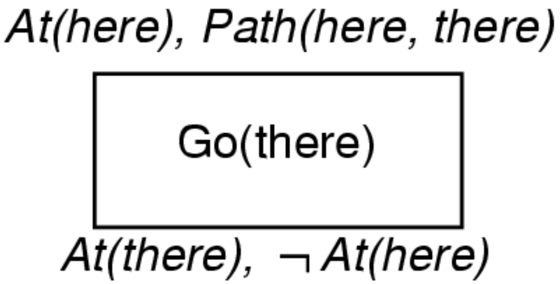
  $At(x) \wedge Sells(x, Milk)$

Blocks World

- Robot Gripper
- Square objects on a perfect, flat table
- Predicates to describe world:
  - On(x,y)
  - TopClear(x)
  - Grip(x)



STRIPS Operators



At(here), Path(here, there)

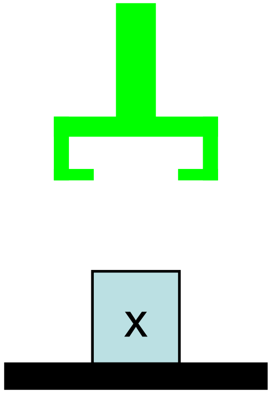
Go(there)

At(there), ¬ At(here)

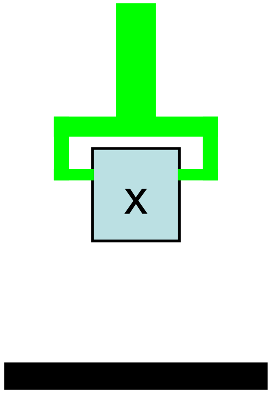
- Operators are triplets of descriptors
  - A set of Preconditions
  - A set of Additions
  - A set of Deletions
- Graphical Representation

Blocks World Operations

- Pickup(x)
  - P: $\mathrm{grip}(\emptyset) \wedge \mathrm{topclear}(x) \wedge \mathrm{on}(x,\mathrm{Table})$
  - A: $\mathrm{grip}(x)$
  - D: $\mathrm{on}(x,\mathrm{Table}) \wedge \mathrm{grip}(\emptyset) \wedge \mathrm{topclear}(x)$

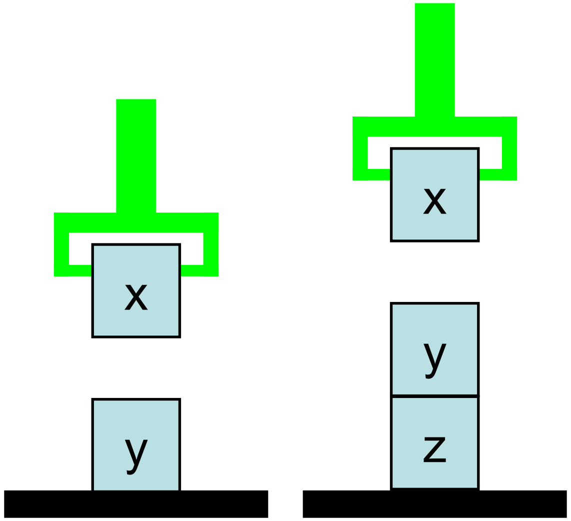


- PutDown(x)
  - P: $\mathrm{grip}(x)$
  - A: $\mathrm{on}(x,\mathrm{Table}) \wedge \mathrm{grip}(\emptyset) \wedge \mathrm{topclear}(x)$
  - D: $\mathrm{grip}(x)$

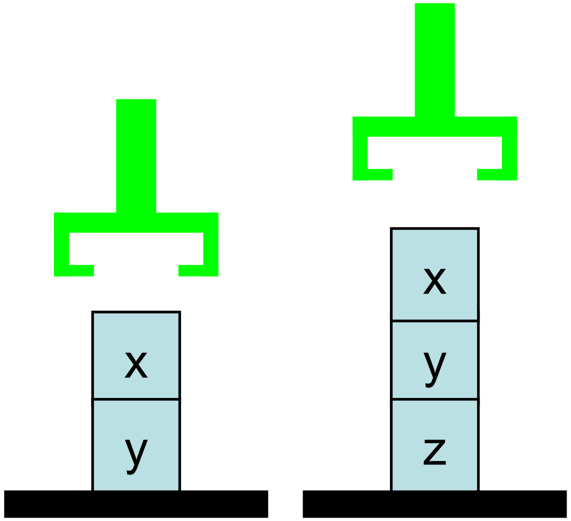


Blocks World Operators II

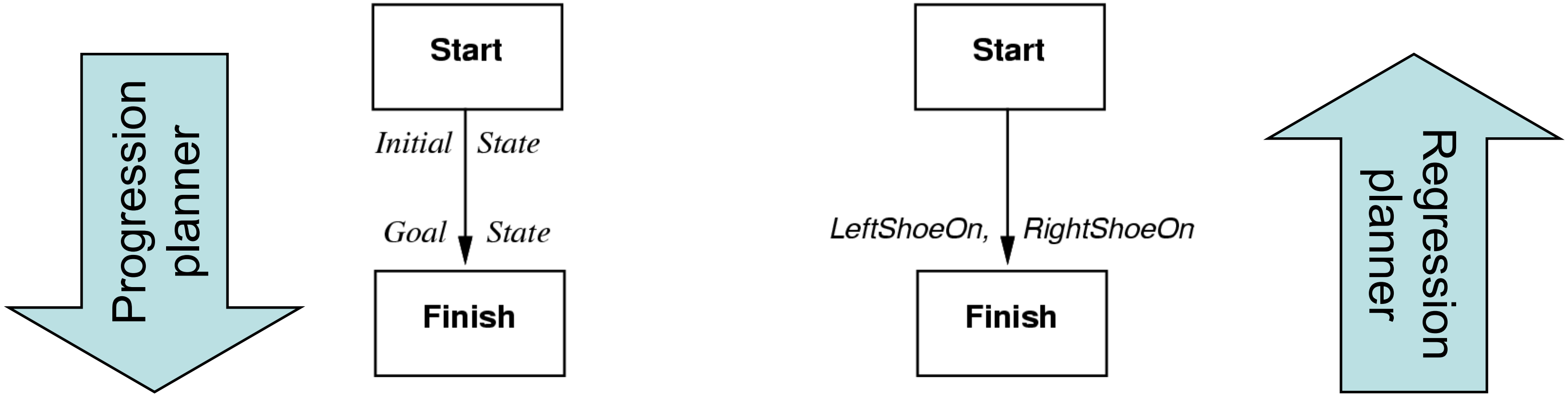
- Stack(x,y)
  - P: topclear(y) ∧ grip(x)
  - A: on(x,y) ∧ grip(∅) ∧ topclear(x)
  - D: topclear(y) ∧ grip(x)



- UnStack(x,y)
  - P: topclear(x) ∧ grip(∅) ∧ on(x,y)
  - A: grip(x) ∧ topclear(y)
  - D: grip(∅) ∧ on(x,y) ∧ topclear(x)



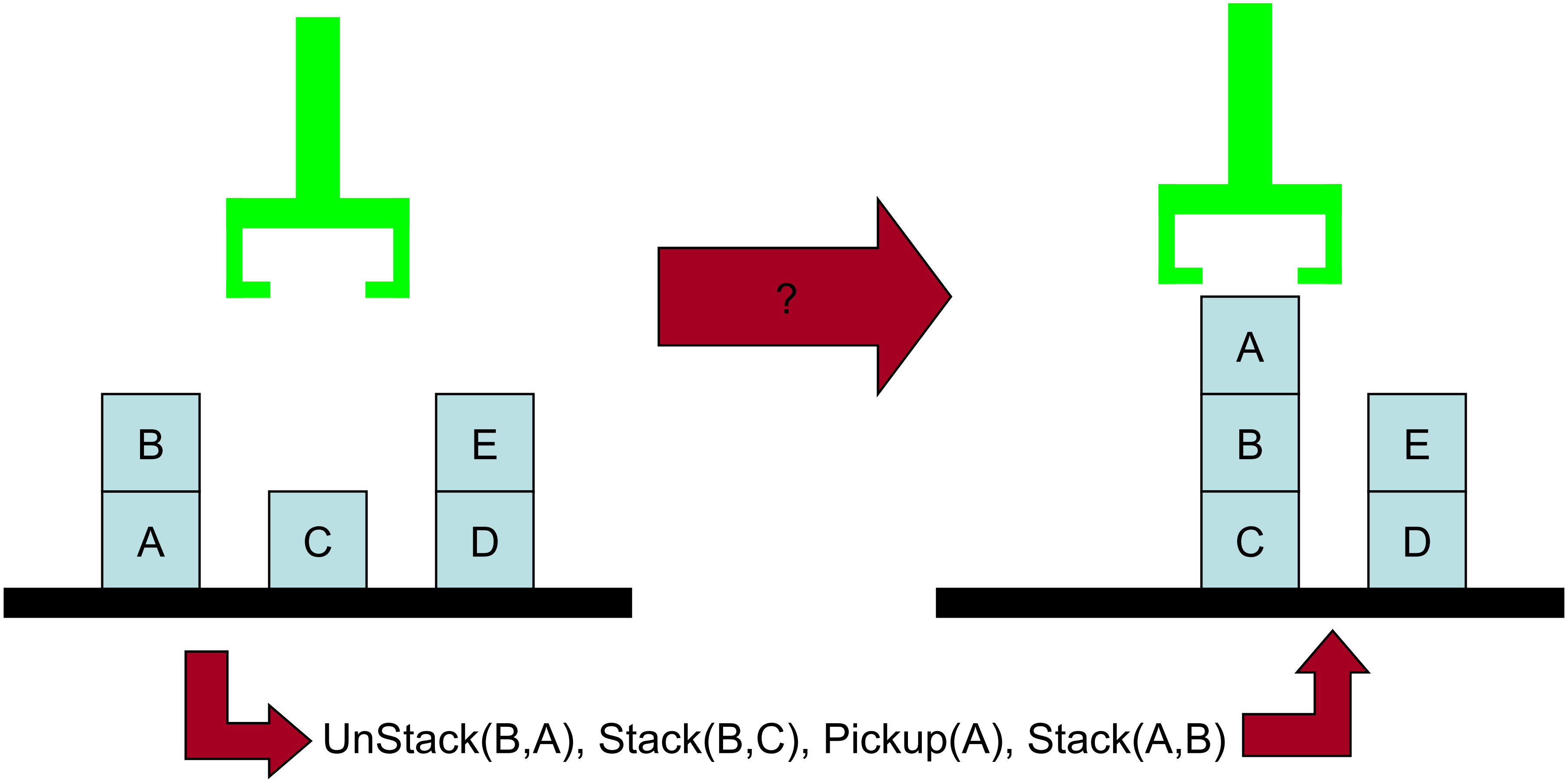
Start and Finish States



(a)　(b)

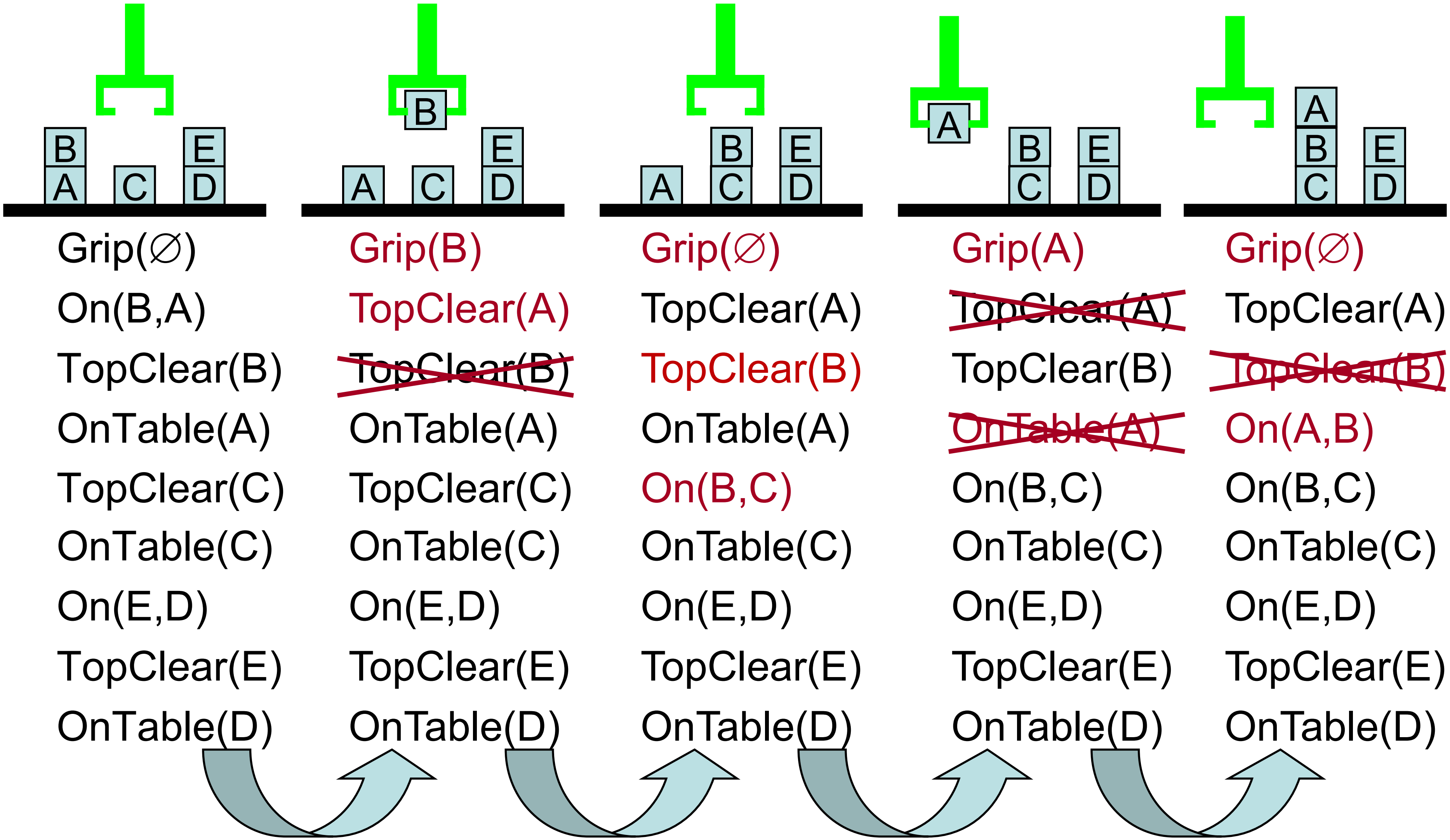
- ***Start*** has no preconditions, and must precede all other actions
- ***Finish*** has no post-conditions, and must not be followed by any other actions
- ***Progressive*** and ***Regressive*** planners

Planning in Blocks World



UnStack(B,A), Stack(B,C), Pickup(A), Stack(A,B)

Update of Knowledge



Grip(∅)	Grip(B)	Grip(∅)	Grip(A)	Grip(∅)
On(B,A)	TopClear(A)	TopClear(A)	~~TopClear(A)~~	TopClear(A)
TopClear(B)	~~TopClear(B)~~	TopClear(B)	TopClear(B)	~~TopClear(B)~~
OnTable(A)	OnTable(A)	OnTable(A)	~~OnTable(A)~~	On(A,B)
TopClear(C)	TopClear(C)	On(B,C)	On(B,C)	On(B,C)
OnTable(C)	OnTable(C)	OnTable(C)	OnTable(C)	OnTable(C)
On(E,D)	On(E,D)	On(E,D)	On(E,D)	On(E,D)
TopClear(E)	TopClear(E)	TopClear(E)	TopClear(E)	TopClear(E)
OnTable(D)	OnTable(D)	OnTable(D)	OnTable(D)	OnTable(D)

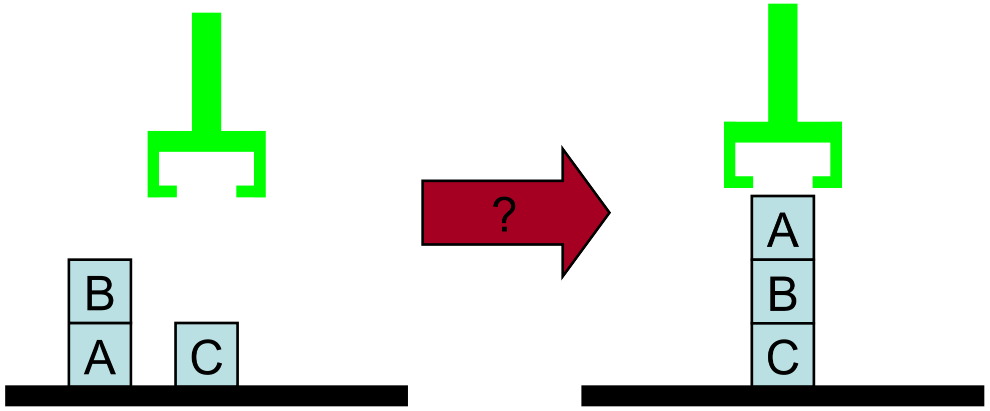
UnStack(B,A)

Stack(B,C)

Pickup(A)

Stack(A,B)

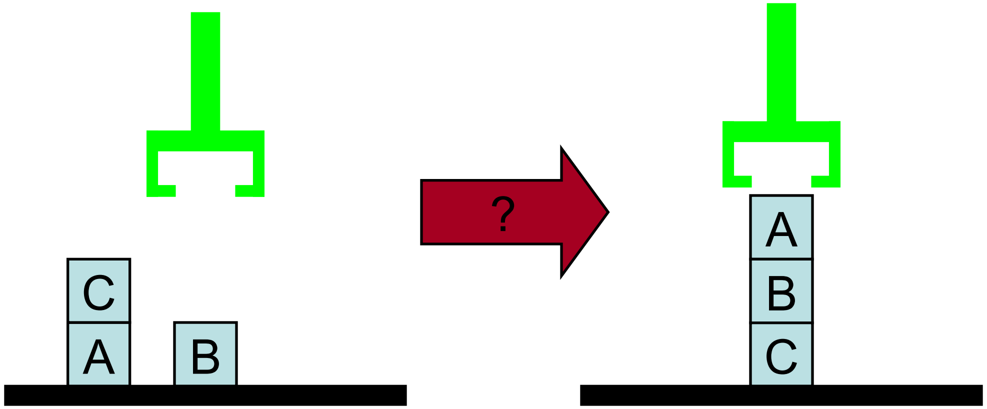
Using Divide and Conquer: Non-interleaved Planners



Sub-goals:
On(B,C)
On(A,B)

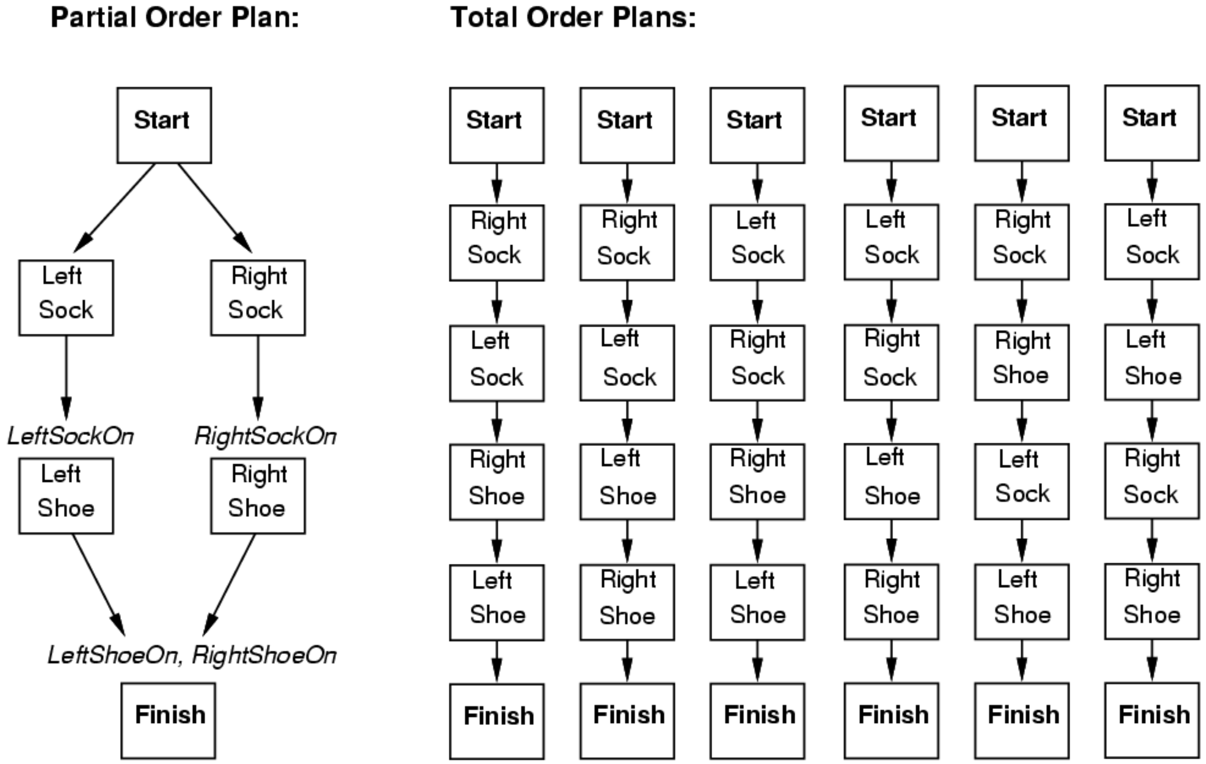
- Divide the problem into subgoals (conjuncts in the goal state) and then plan each of those individually
- Goal : On(B,C) $\wedge$ On(A,B)
- Perform all of the actions to accomplish goal #1, then all the actions to accomplish goal #2, etc.

Problems with Non-interleaved Planners



- Switch the location of B and C in the initial state
- Same Goal : On(B,C) ∧ On(A,B)
- If you attempt to solve either one of these first, then you cannot solve the second without breaking the first solution
- Sussman Anomaly

Representations for Plans

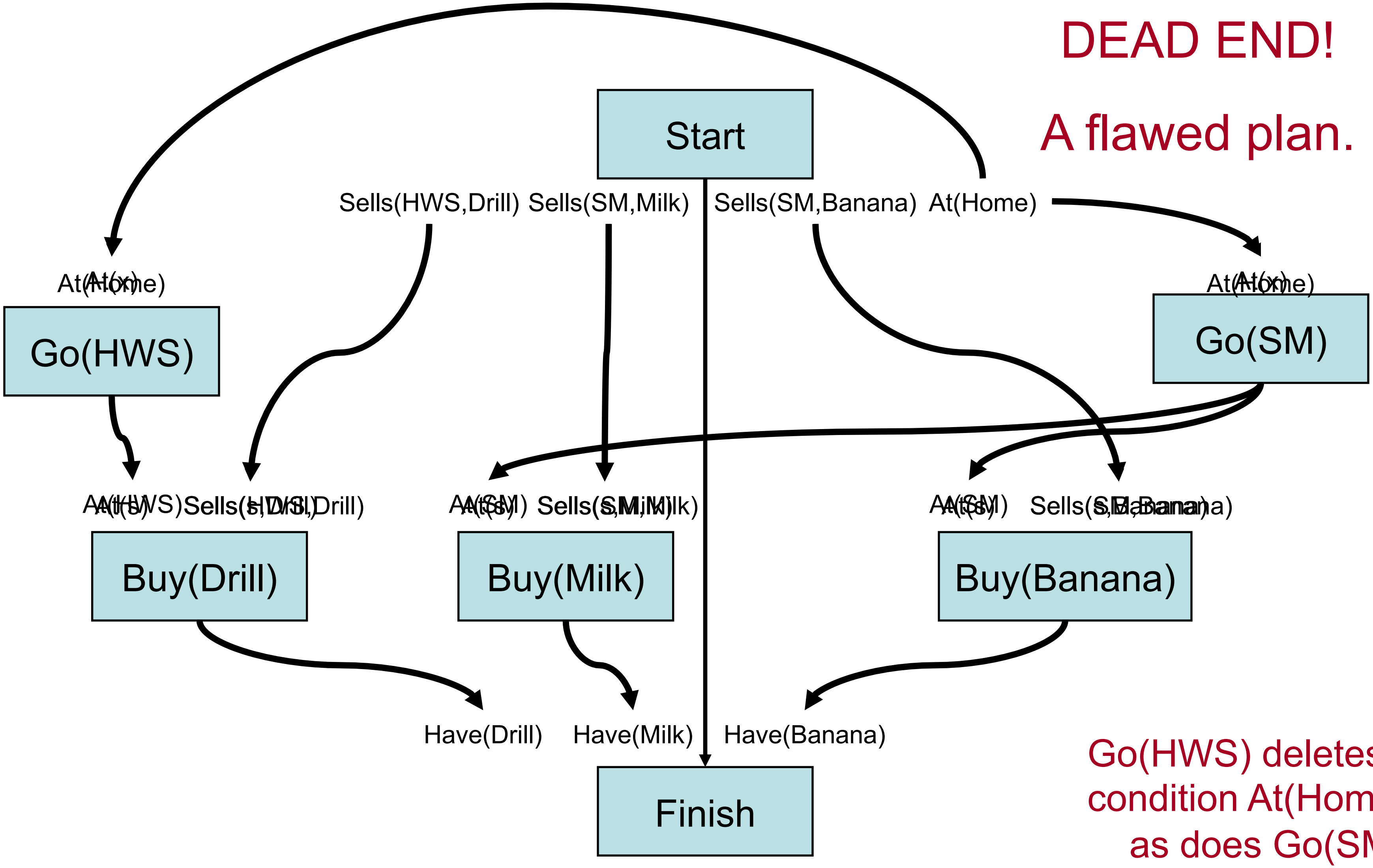


- Causal links (heavy arrows) : achieve a required prerequisite
- Ordering links (light arrows) : semi-arbitrary ordering
- Principle of Least Committment

Partial Order Planning

DEAD END!

A flawed plan.



Start

Sells(HWS,Drill) Sells(SM,Milk) Sells(SM,Banana) At(Home)

At(Home)
At(x)

Go(HWS)

At(Home)
At(x)

Go(SM)

At(HWS) Sells(HWS,Drill)
At(s) Sells(s,Drill)

Buy(Drill)

At(SM) Sells(SM,Milk)
At(s) Sells(s,Milk)

Buy(Milk)

At(SM) Sells(SM,Banana)
At(s) Sells(s,Banana)

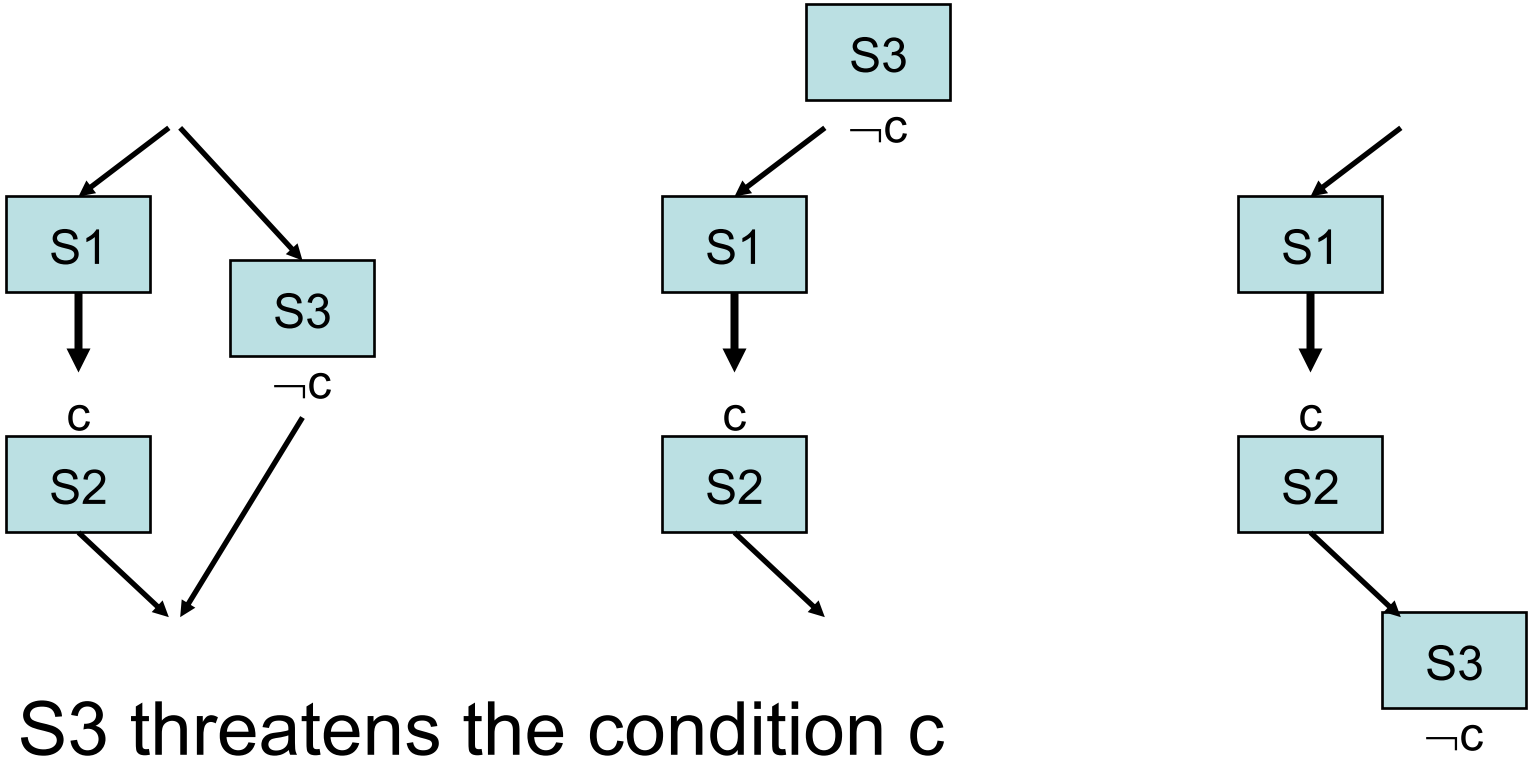
Buy(Banana)

Have(Drill) Have(Milk) Have(Banana)

Finish

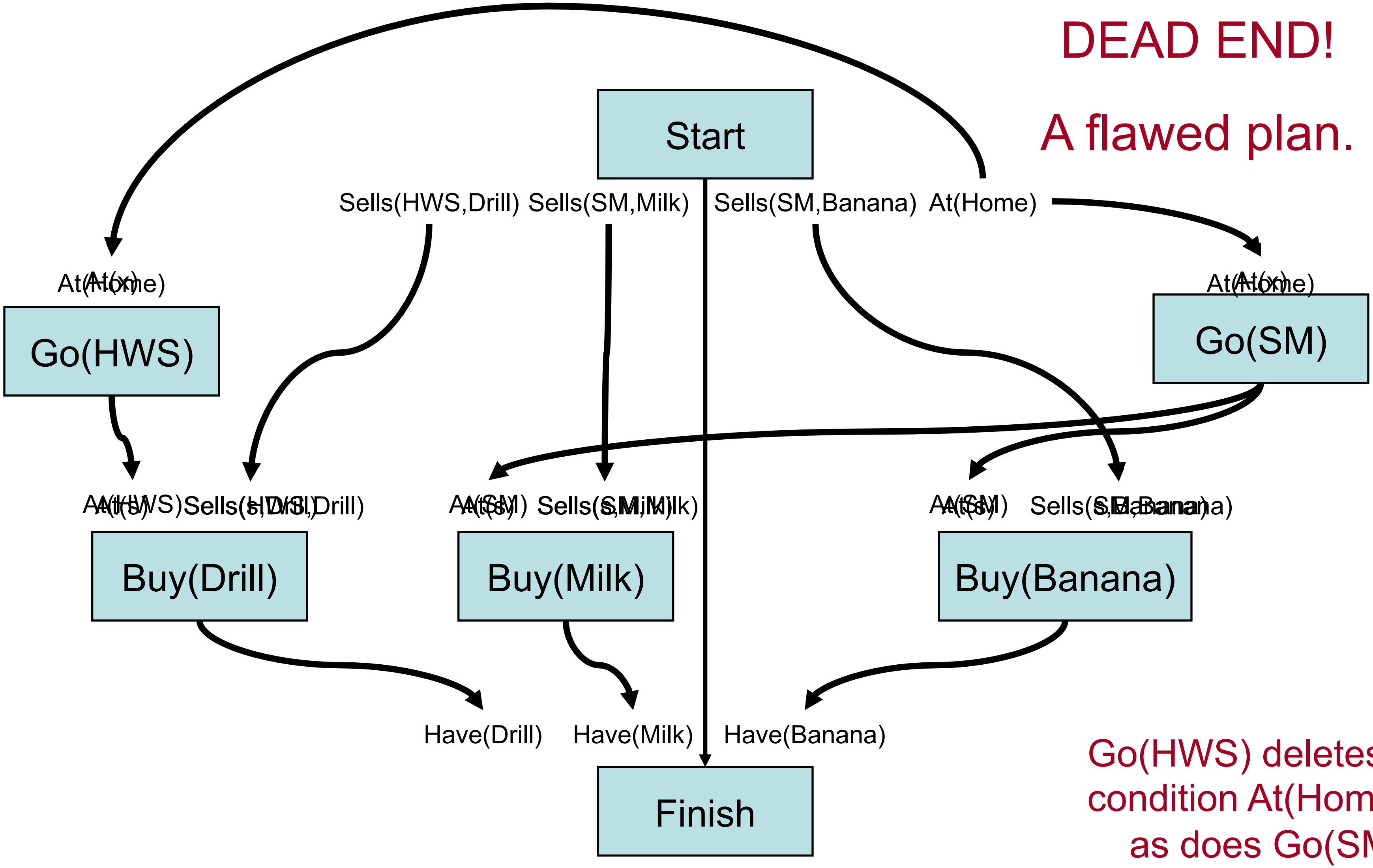
Go(HWS) deletes the condition At(Home)… as does Go(SM)

Protecting Causal Links



- S3 threatens the condition c
- S3 can then either be demoted or promoted to avoid the conflict

Partial Order Planning



Start

Sells(HWS,Drill) Sells(SM,Milk) Sells(SM,Banana) At(Home)

At(Home) At(x)

Go(HWS)

At(Home) At(x)

Go(SM)

At(HWS) At(s) Sells(HWS,Drill) Sells(s,Drill)

Buy(Drill)

At(SM) At(s) Sells(SM,Milk) Sells(s,Milk)

Buy(Milk)

At(SM) At(s) Sells(SM,Banana) Sells(s,Banana)

Buy(Banana)

Have(Drill) Have(Milk) Have(Banana)

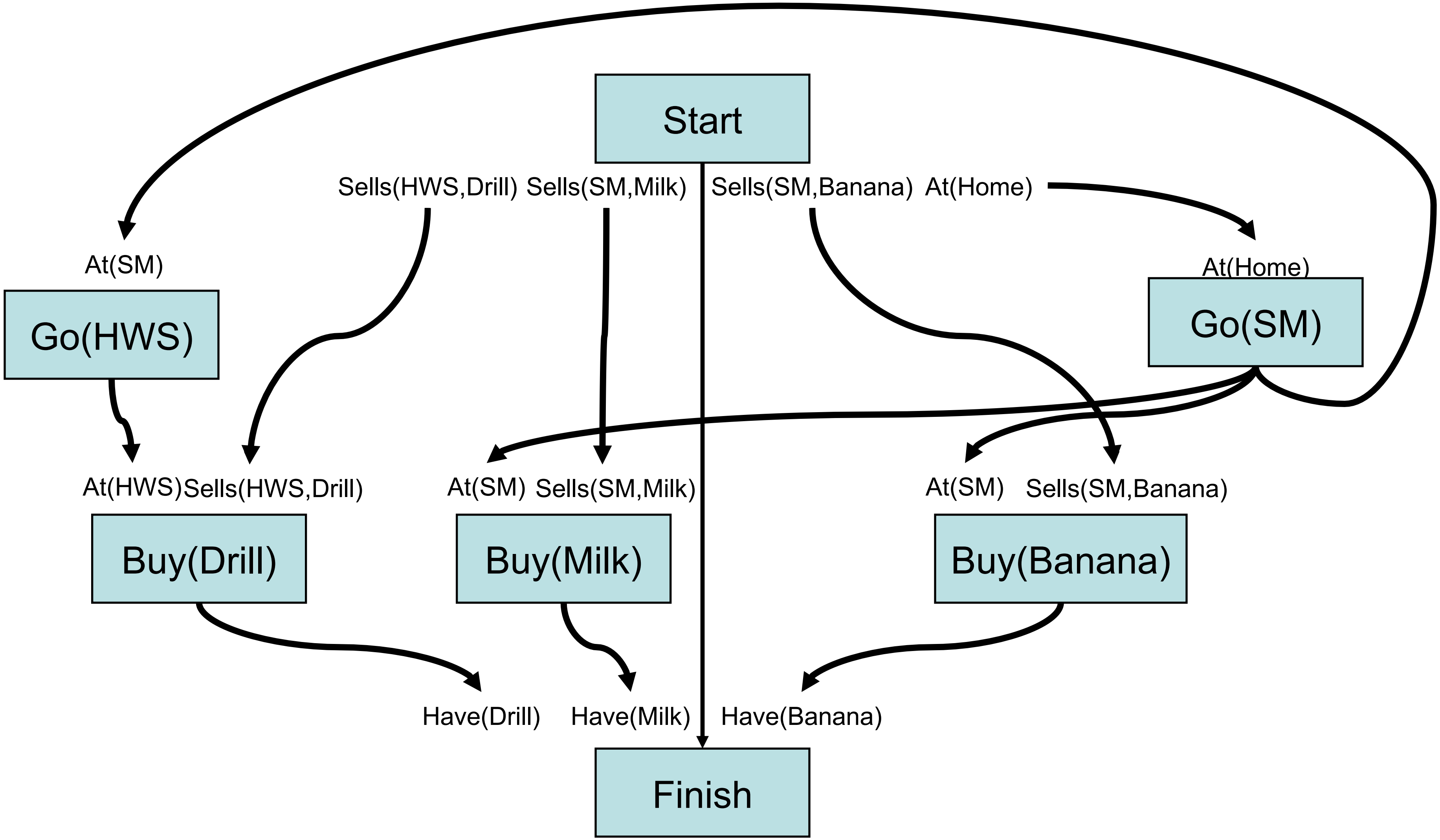
Finish

DEAD END!

A flawed plan.

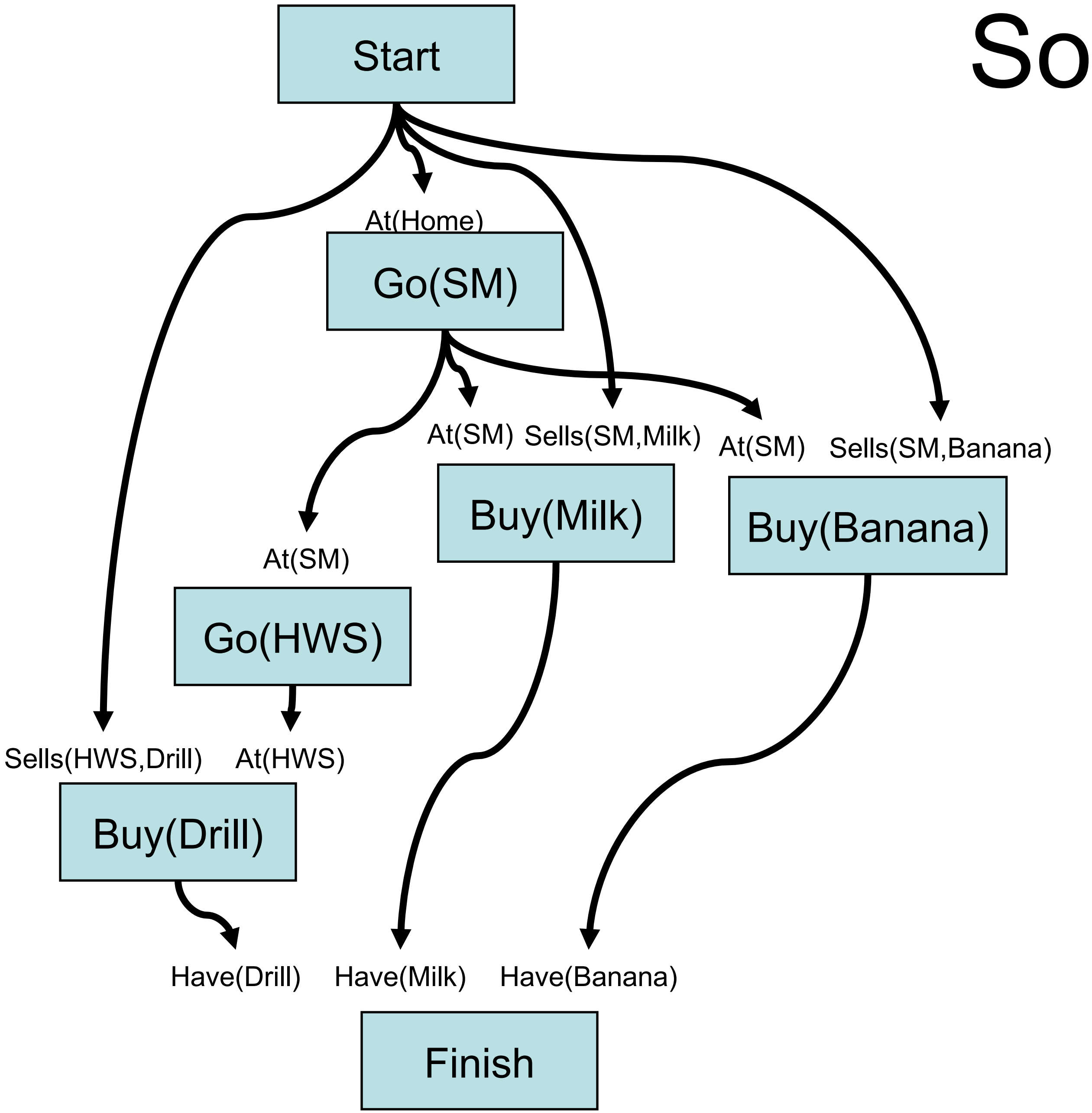
Go(HWS) deletes the
condition At(Home)…
as does Go(SM)

Back to our Example…



Solution

- Showing only causal links

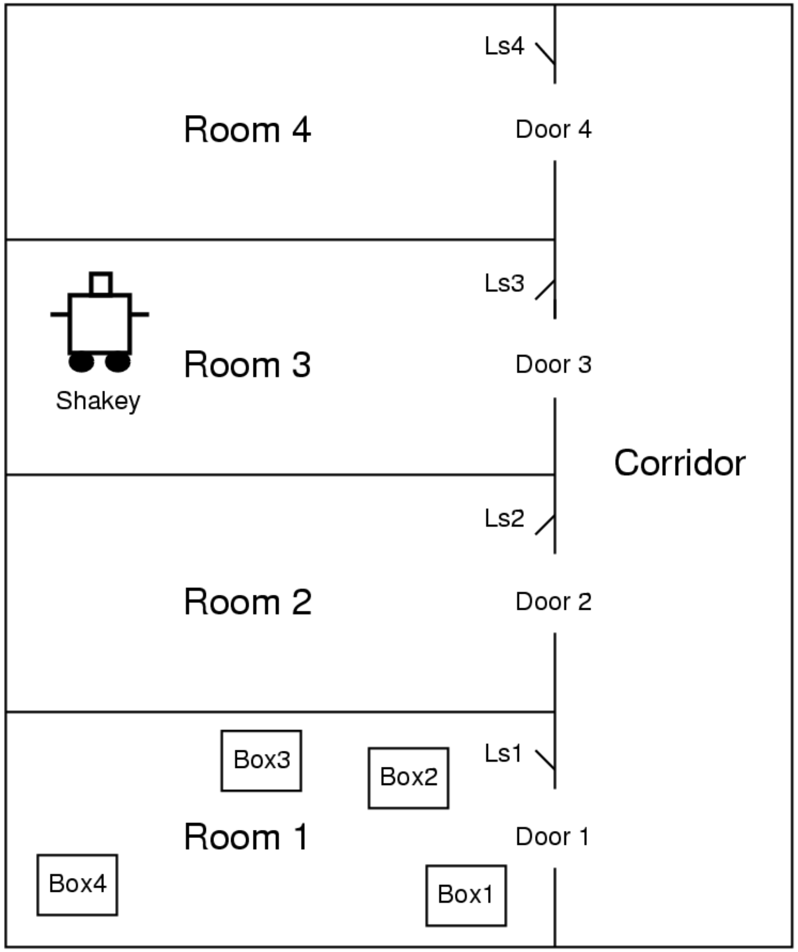


Planning with Existentials and Possible Threats

- Is $\neg$At(x) a threat to At(Home) ?
- Depends on what x is later instantiated to be…it could possibly be a threat
- Ways to treat possible threats
  - Resolve now with equality (assign x to be something other than Home right now)
  - Resolve now with inequality (introduce the constraint x≠Home)
  - Resolve Later

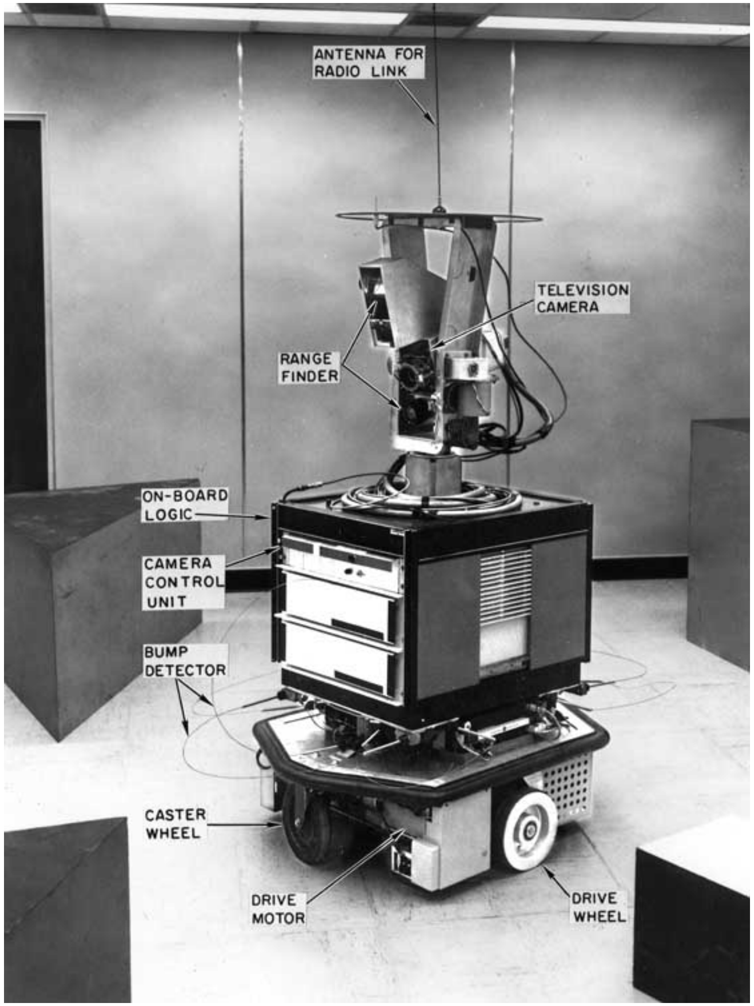
Shakey the Robot

- Demonstration of STRIPS planning
- Known set of rooms and a connecting corridor
- Light switches could only be reached by climbing on a box



Shakey the Robot (SRI, 1970)

- Sensors
  - Laser Range finder
  - Television Camera
  - Bump Sensors
- Off-board computation
- Environment
  - Static
  - Controlled lighting
  - Polyhedral solids
  - Solid colors



Special Guest Lecture Monday

From Soup to Nuts: Using AI and Robotics to Improve Human Robot Interaction

Greg Trafton

Naval Research Laboratory

How do experts interact in dynamic, dangerous domains, and how can AI improve their interaction? We studied how expert Navy firefighters worked together to put out shipboards fires and how they interacted. We show how an unexpected finding lead to a novel approach for interaction.