

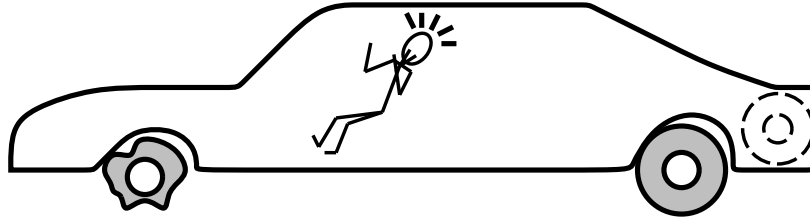
PLANNING AND ACTING

CHAPTER 12

Outline

- ◇ The real world
- ◇ Conditional planning
- ◇ Monitoring and replanning

The real world



START

*~Flat(Spare) Intact(Spare) Off(Spare)
On(Tire1) Flat(Tire1)*

On(x) ~Flat(x)

FINISH

On(x)

Remove(x)

Off(x) ClearHub

Off(x) ClearHub

Puton(x)

On(x) ~ClearHub

Intact(x) Flat(x)

Inflate(x)

~Flat(x)

Things go wrong

Incomplete information

Unknown preconditions, e.g., *Intact(Spare)*?

Disjunctive effects, e.g., *Inflate(x)* causes

$\text{Inflated}(x) \vee \text{SlowHiss}(x) \vee \text{Burst}(x) \vee \text{BrokenPump} \vee \dots$

Incorrect information

Current state incorrect, e.g., spare NOT intact

Missing/incorrect postconditions in operators

Qualification problem:

can never finish listing all the required preconditions and possible conditional outcomes of actions

Solutions

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

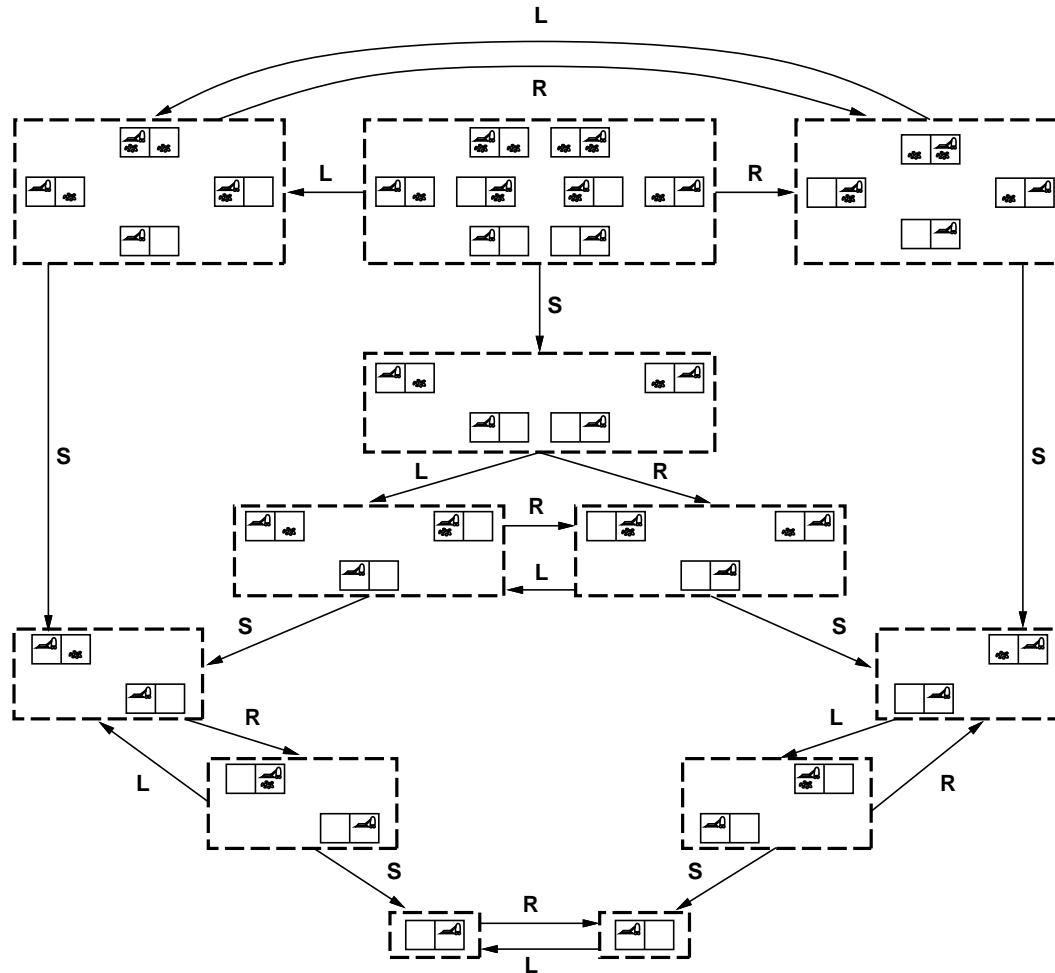
Check progress *during execution*, replan if necessary

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)

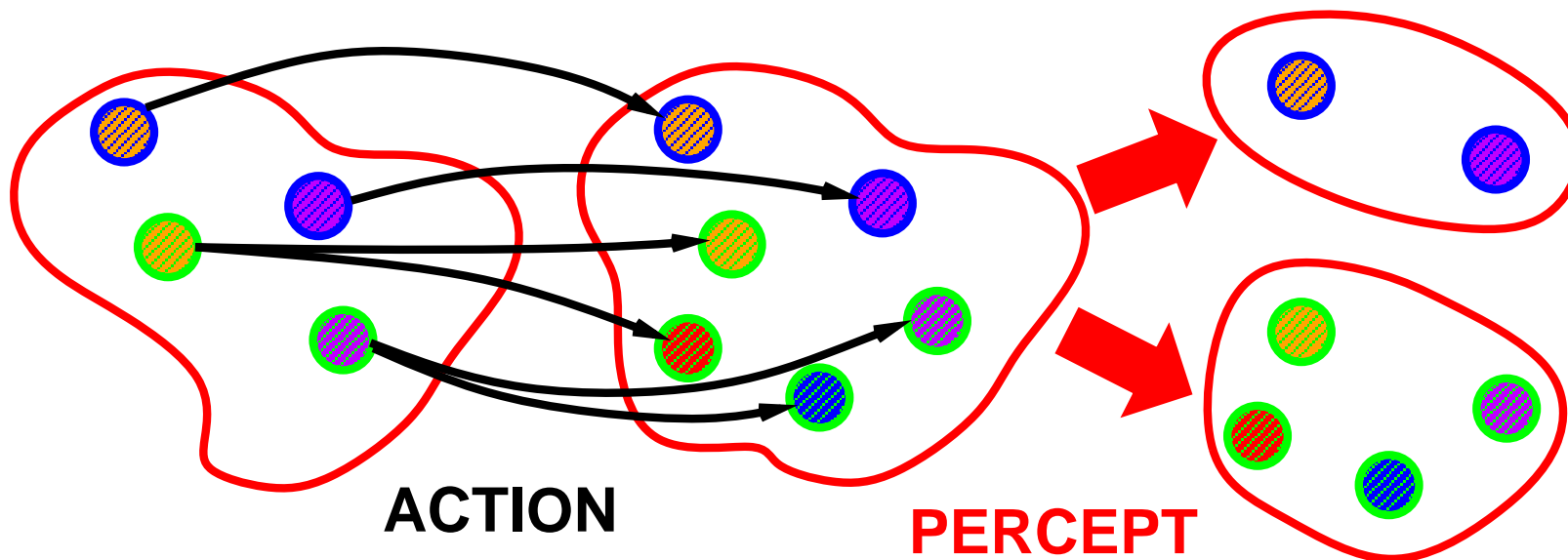
Conformant planning

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



Conditional planning

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



Conditional planning contd.

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

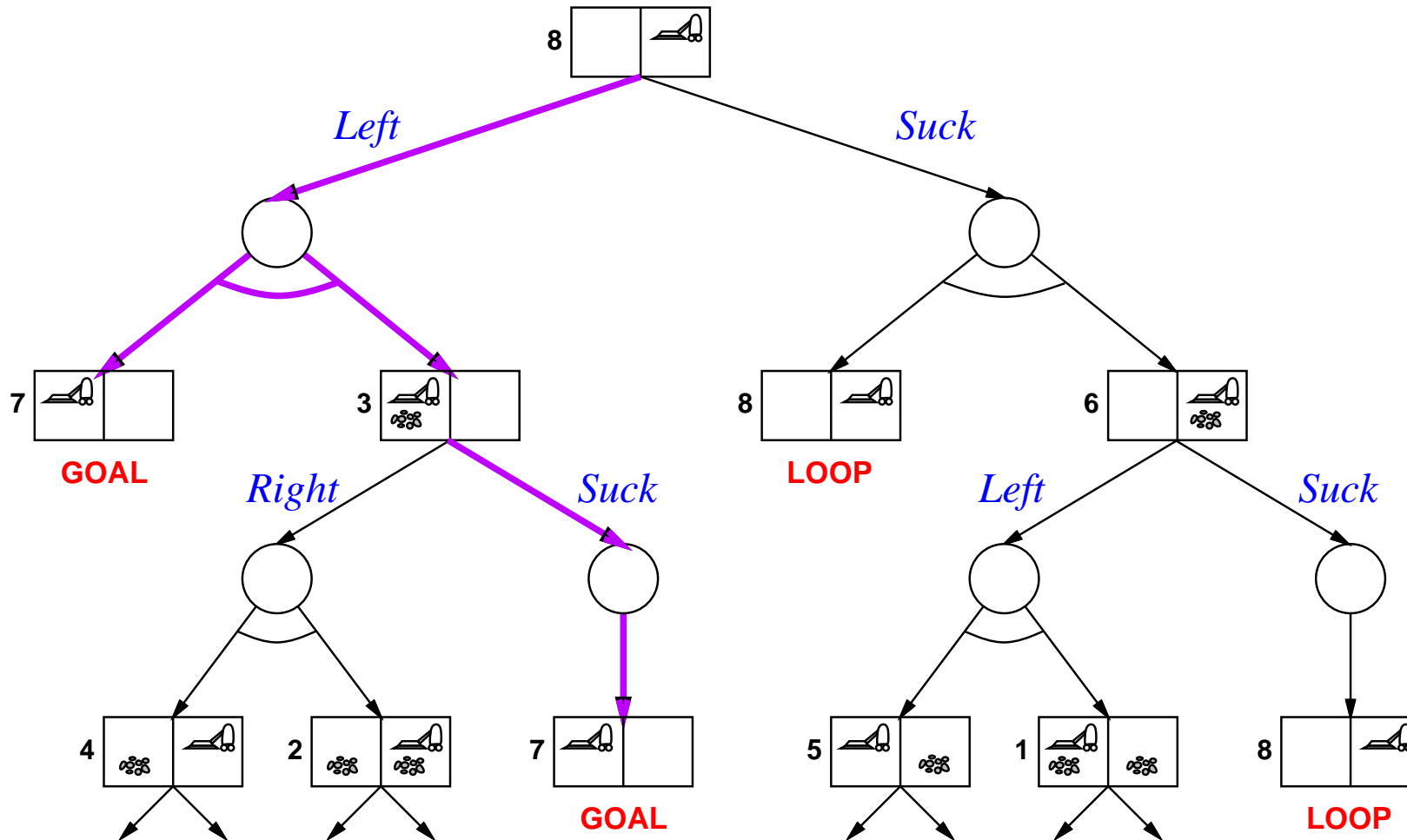
(Cf. game playing: *some* response for *every* opponent move)

(Cf. backward chaining: *some* rule such that *every* premise satisfied)

AND–OR tree search (very similar to backward chaining algorithm)

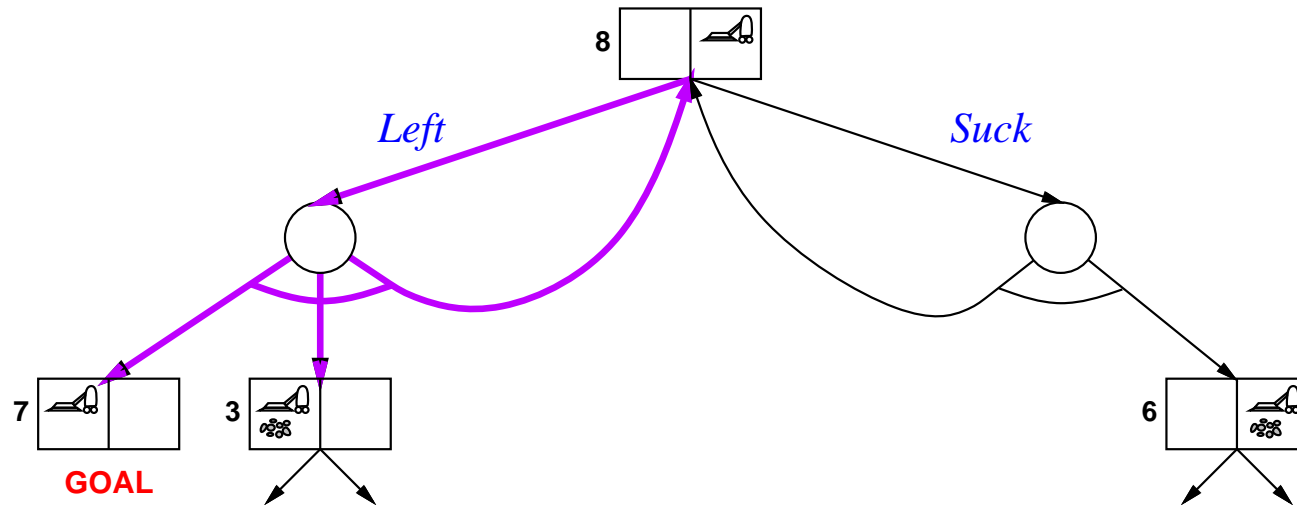
Example

Double Murphy: sucking or arriving may dirty a clean square



Example

Triple Murphy: also sometimes stays put instead of moving



$[L_1 : \text{Left, if } AtR \text{ then } L_1 \text{ else [if } CleanL \text{ then } [] \text{ else } Suck]]$

or $[\text{while } AtR \text{ do } [Left], \text{if } CleanL \text{ then } [] \text{ else } Suck]$

“Infinite loop” but will eventually work unless action always fails

Execution Monitoring

“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

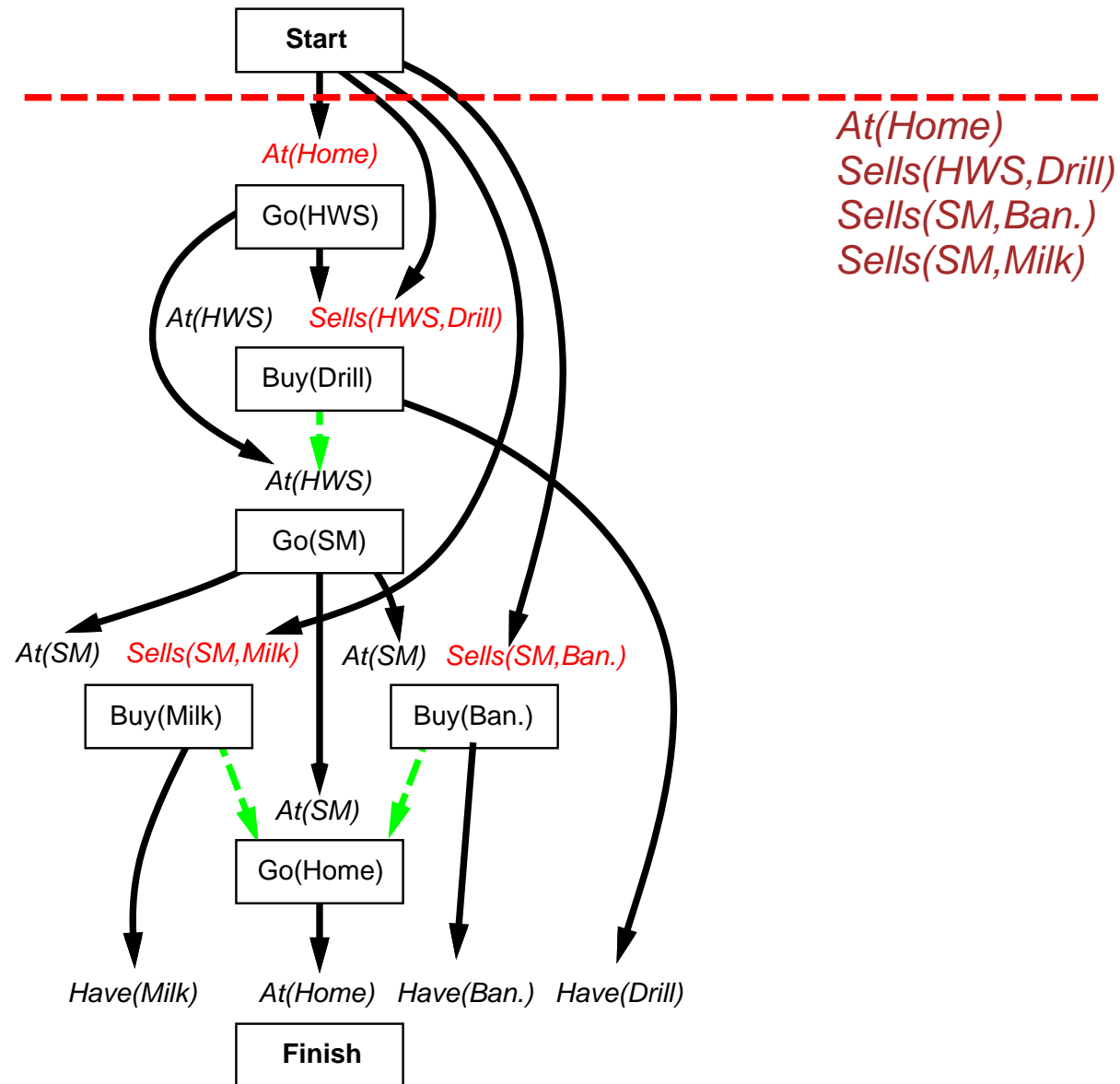
On failure, resume POP to achieve open conditions from current state

IPEM (Integrated Planning, Execution, and Monitoring):

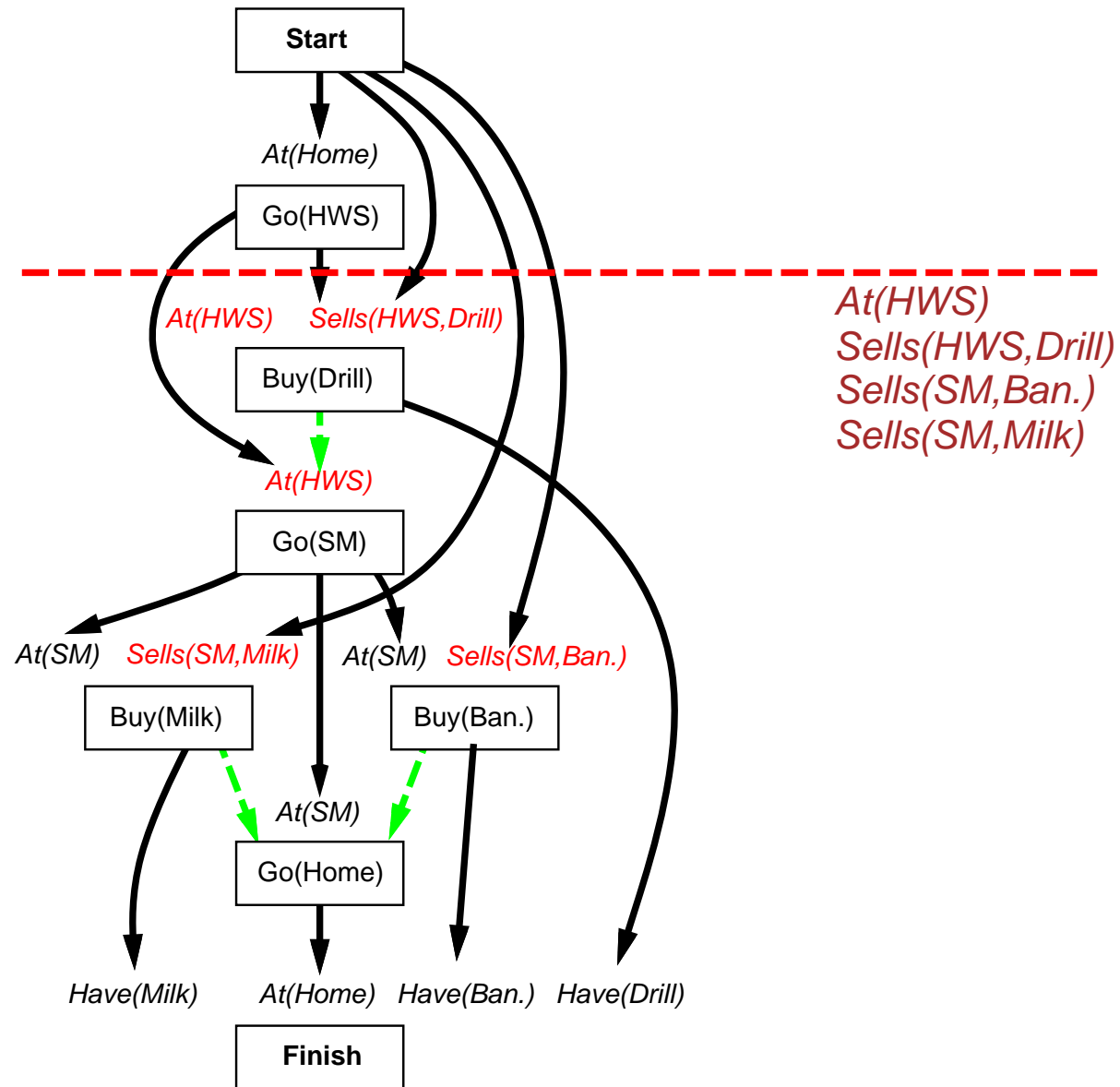
- keep updating *Start* to match current state

- links from actions replaced by links from *Start* when done

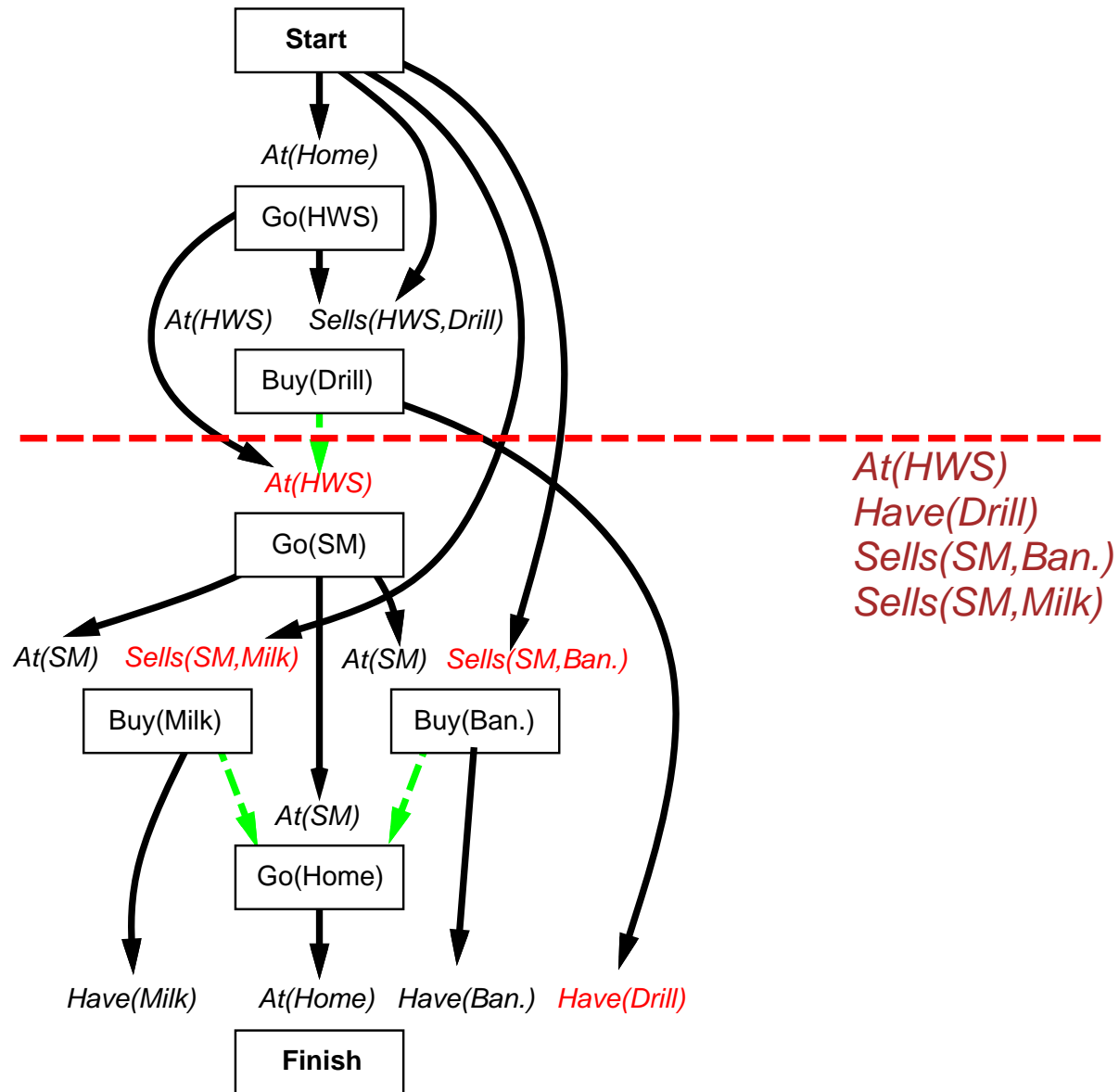
Example



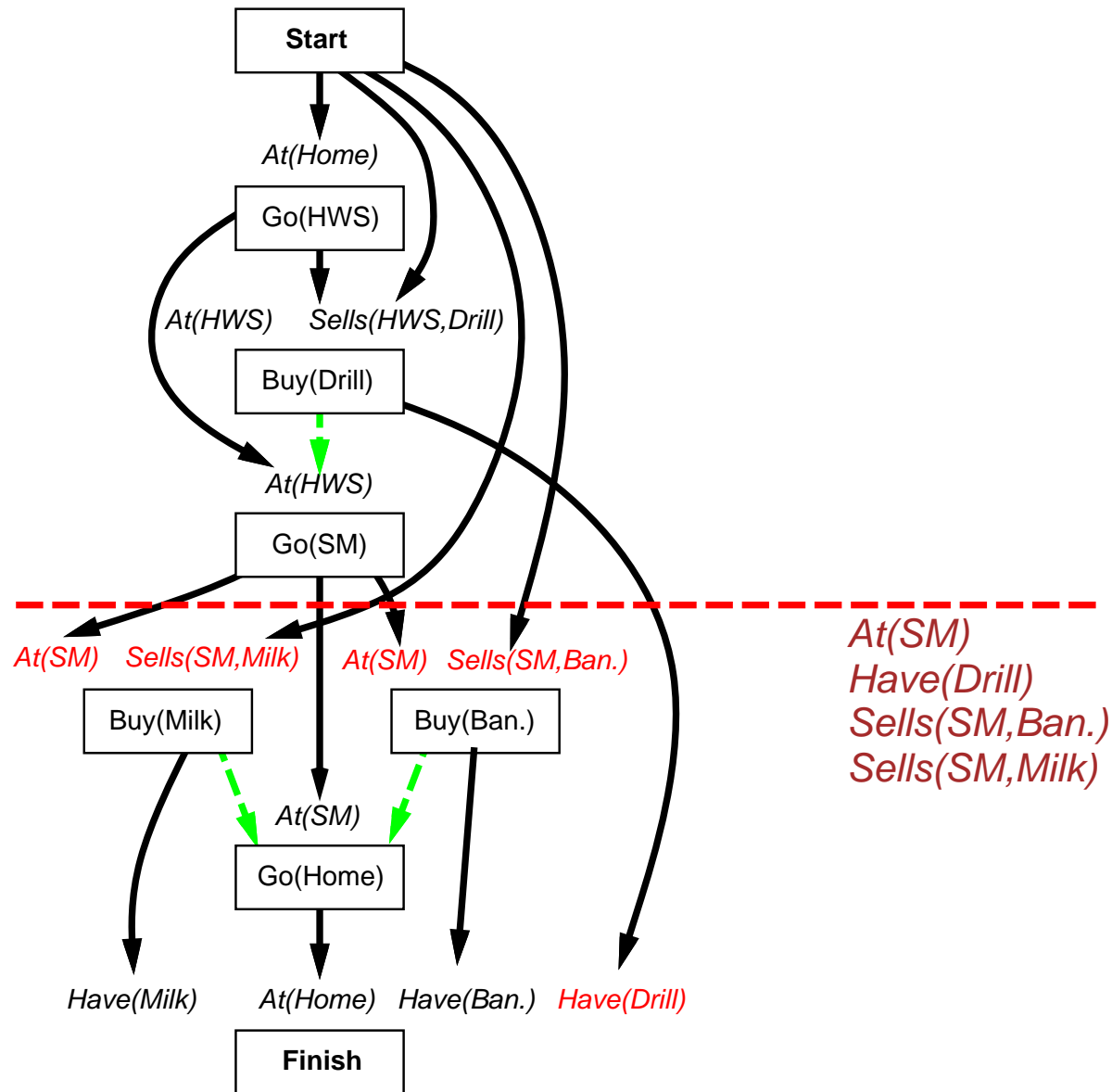
Example



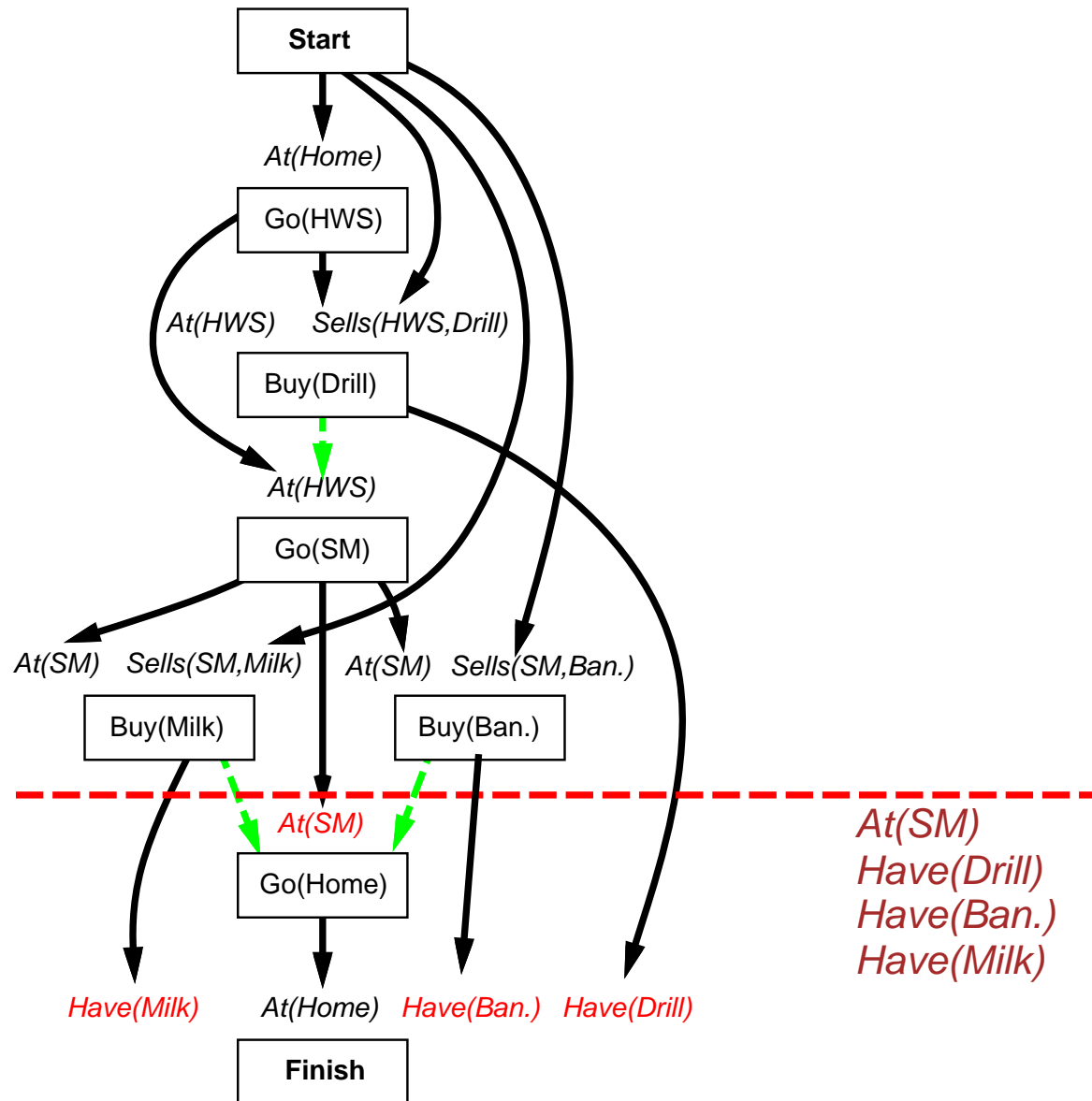
Example



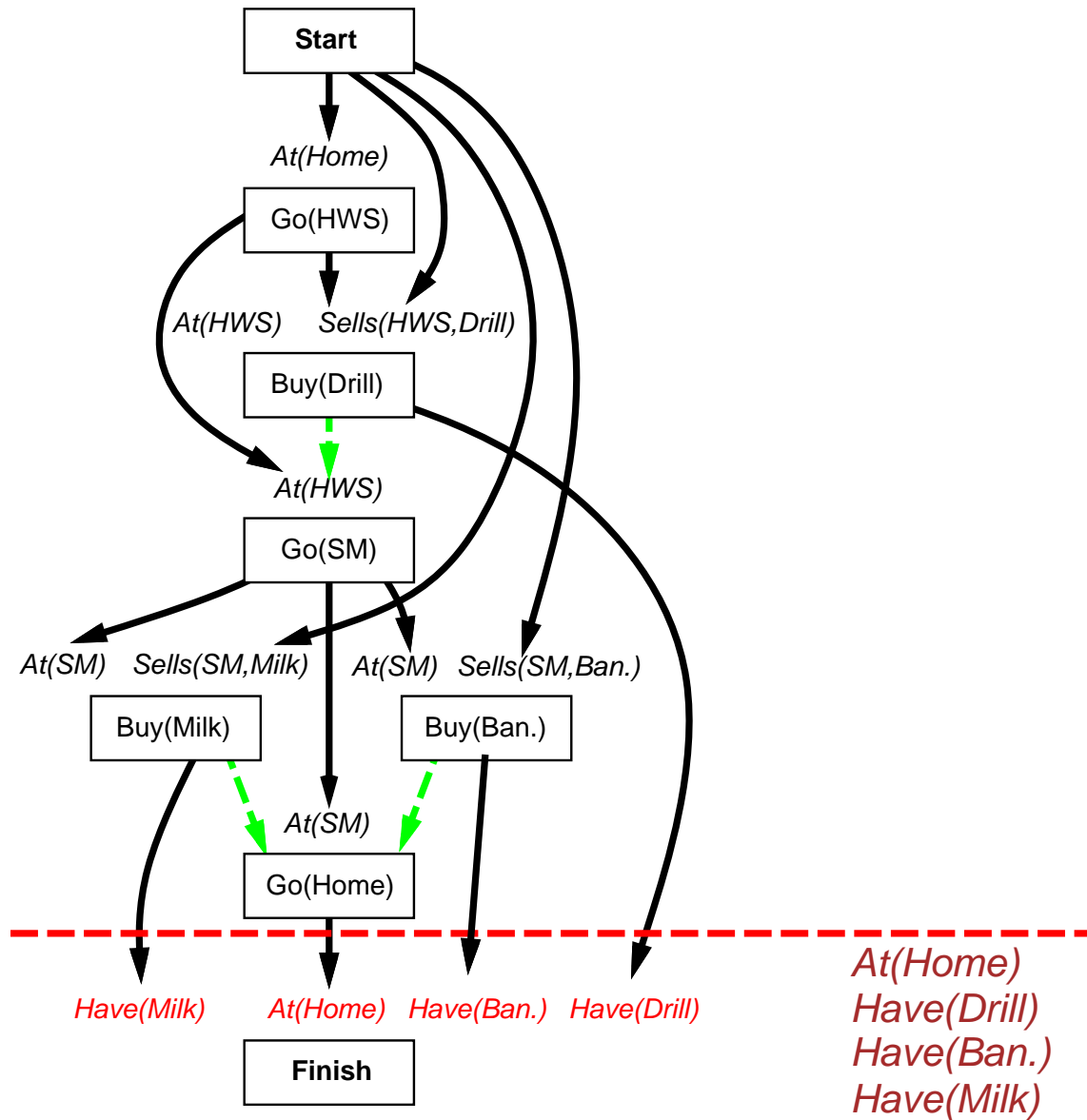
Example



Example



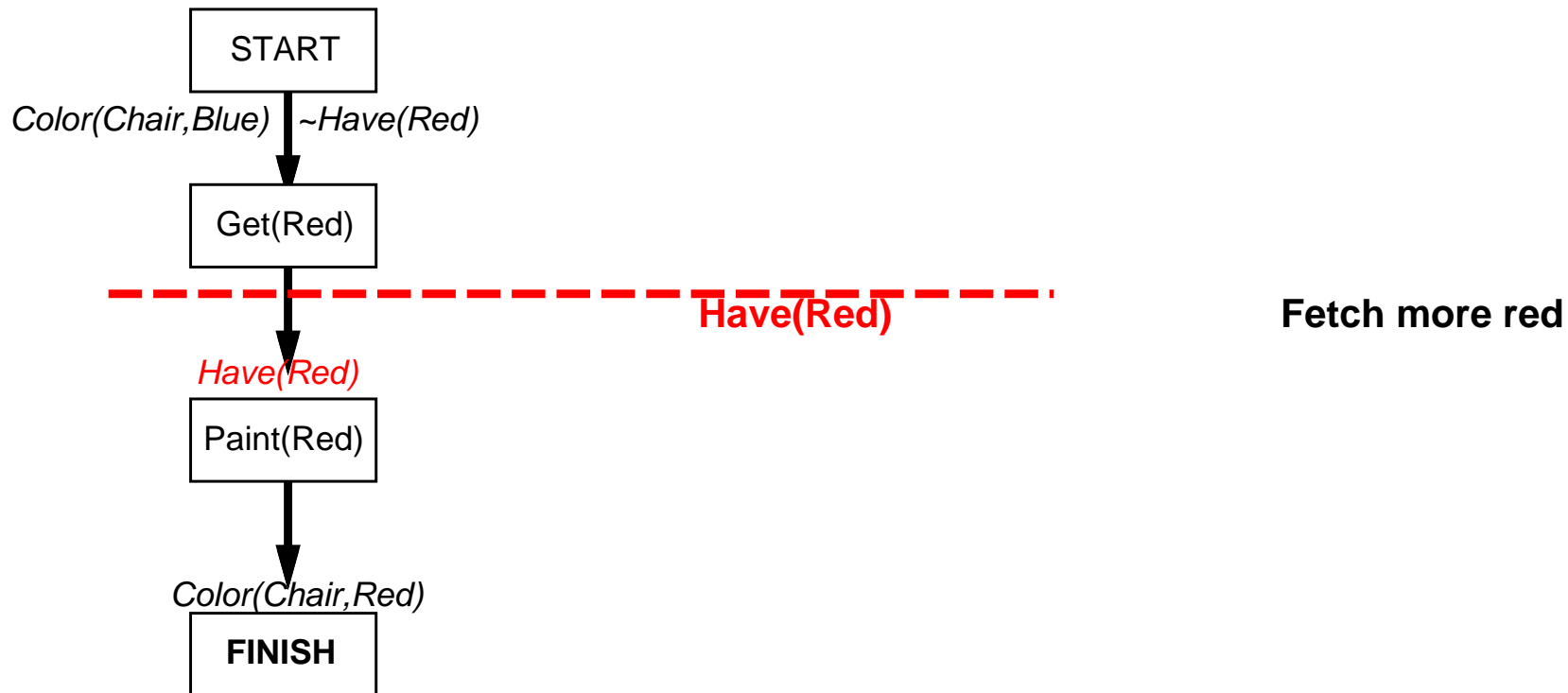
Example



Emergent behavior

PRECONDITIONS

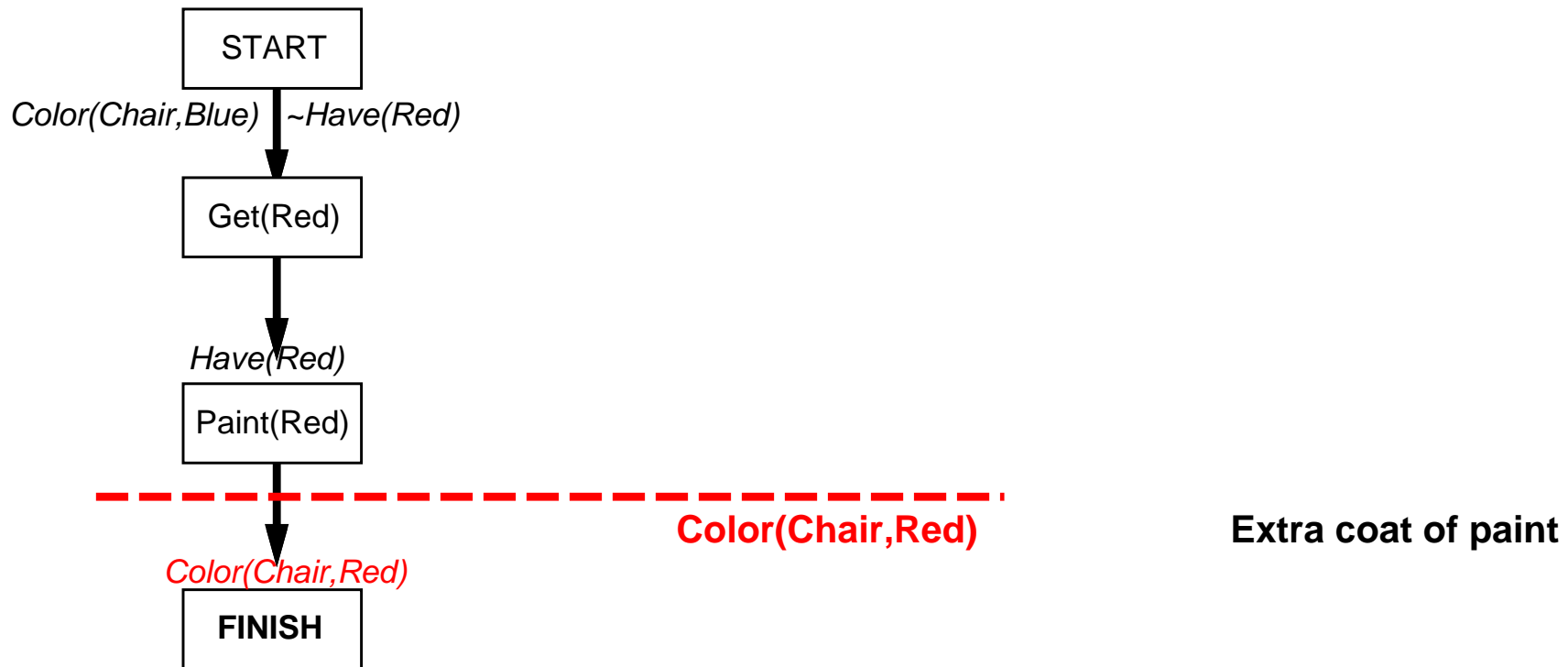
FAILURE RESPONSE



Emergent behavior

PRECONDITIONS

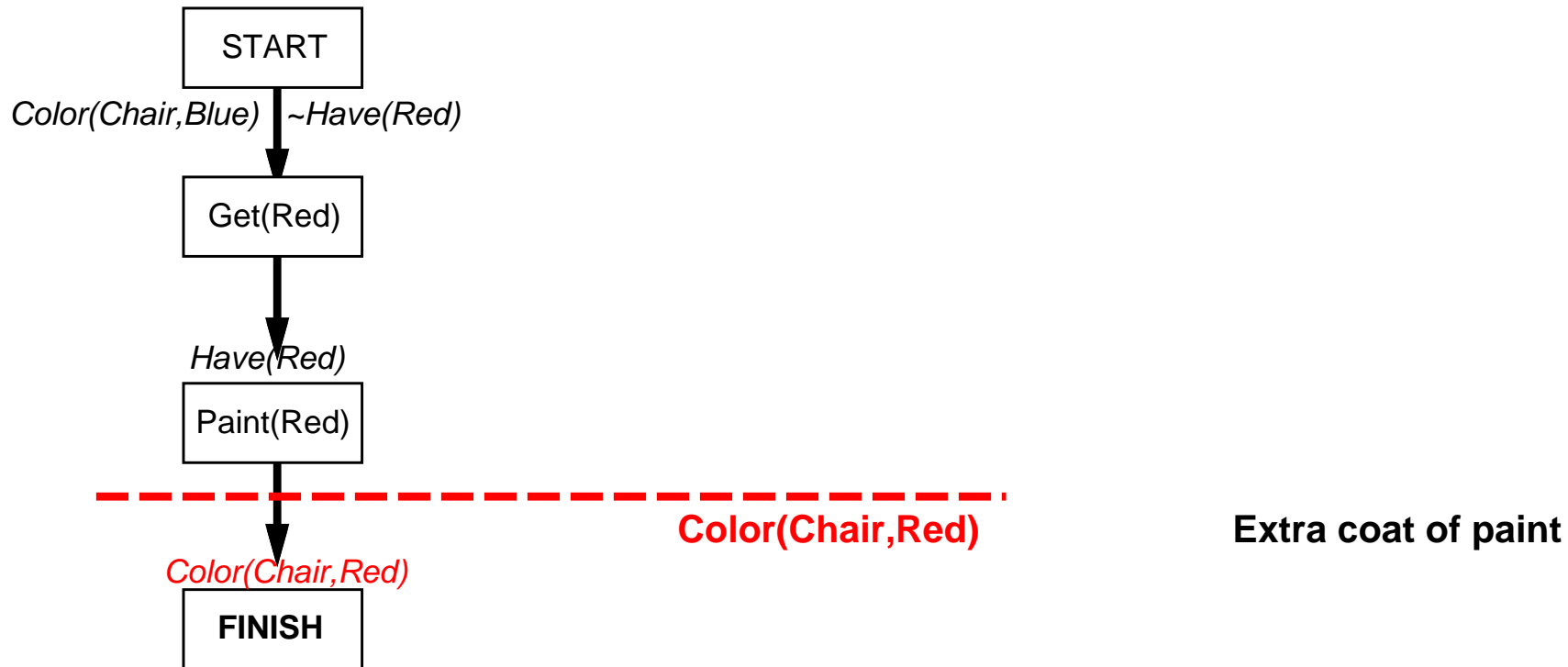
FAILURE RESPONSE



Emergent behavior

PRECONDITIONS

FAILURE RESPONSE



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