Title: Planning & Acting
AIMA: Chapter 13 (Section 13.2-13.3)

Introduction to Artificial Intelligence
CSCE 476-876, Spring 2000
www.cse.unl.edu/~cse476

Berthe Y. Choueiry
choueiry@cse.unl.edu, (402)472-5444

Outline

- Execution vs. action monitoring
- Causes for plan failure
- Strategies for replanning
  - Simple replanning and execution monitoring
  - Fully integrated planning and execution monitoring

World go wrong:

Suppose all preconditions of a step are not met

if the step is executed, state wont be same as predicted

i.e., executing the step yields a failure

remaining plan fails....

— **Execution monitoring**

— **Action monitoring**

(world is deterministic, accessible, but dynamic)

Execution monitoring: standard model

Consider a plan segment,
— consider all preconditions of steps in segment not
established by other steps in segment
— annotate the plan segment with these preconditions

Potential failure detected by comparing
these preconditions with
the state description generated from the percept sequence

Action monitoring:

Check preconditions of each action as it is executed

Action vs. execution monitoring:

+ Simpler
+ No need for annotations
− Less effective (does not look ahead)
+ useful when goal is serendipitously achieved

Note: the world need to be accessible

Causes for plan failure

Bounded indeterminacy :

Actions may have unexpected effects but these can be enumerated

e.g., open a paint can: find paint, empty, spill paint

→ use conditional planning

Unbounded indeterminacy :

Unexpected effects too many to enumerate

Plan for a limited number of contingencies

Re-plan when reality does not behave as expected

Two strategies for replanning

1. Simple replanning and execution monitoring
   get plan back on track as quickly as possible
2. Fully integrated planning and execution monitoring

Simple replanning with....: Principle

- Consider a plan p with steps $S_1, S_2, S_3, \ldots S_n$ being executed
- After S_{i-1}, sensors perceive that a precond of S_i is <u>not</u> established
- Identify current state S_c
- Choose at step S_j in the plan $(S_1, S_2, S_3 \ldots, S_n)$ such that $(S_j, S_{j+1}, \ldots, S_n)$ is the easiest to achieve from S_c,
- Devise a new plan to achieve the preconditions of S_j from the current state S_c
- Execute the plan obtained by concatenating the new plan to $(S_j, S_{j+1}, \ldots, S_n)$

Compare: CPOP vs. simple replanning with..

Example: Given a table and a chair, SenseColor of the table, GetPaint of the color of the table, Paint the chair

Simple replanning:

1- Suppose painting is imperfect and leaves empty spots on chair

After painting the chair, checks whether preconditions of the rest of the plan are met, which are....... ?

Failure detection..

Table not same color as chair, decides to repaint, will cycle until chair is completely painted

2- Suppose agent run out of paint

Failure detection..

Current state matches preconditions of GetPaint

Agent goes off get new can, paints..

Conditional planner:

Reasons explicitly about possibility of uneven paint

Prepares for it even though it may not occur

Will have to build a looping plan

Comparison:

– Identical: Effective behavior

– Differ: computation carried out and information available to the computation process

Fully integrated Planning and execution

So far, planner and the execution monitor are separate processes
Now, they will be integrated

Planner's activities:

- execute some steps of the plan ready to be executed
- refine plan to resolve deficiencies
  e.g., open condition, potential clobber, etc.
- refine plan in light of new information obtained during execution
- fix plan in light of unexpected changes in environment
  e.g., recover from execution errors, remove steps that have become unnecessary by serendipitous occurrences
- ..

Planner's strategy:

- First spend some time to build a plan
- Then, start execution before plan is complete, especially when there are independent subgoals continuously monitor the world
- Update world model, even if deliberations are continuing

Example: Block world

Start state: (a)

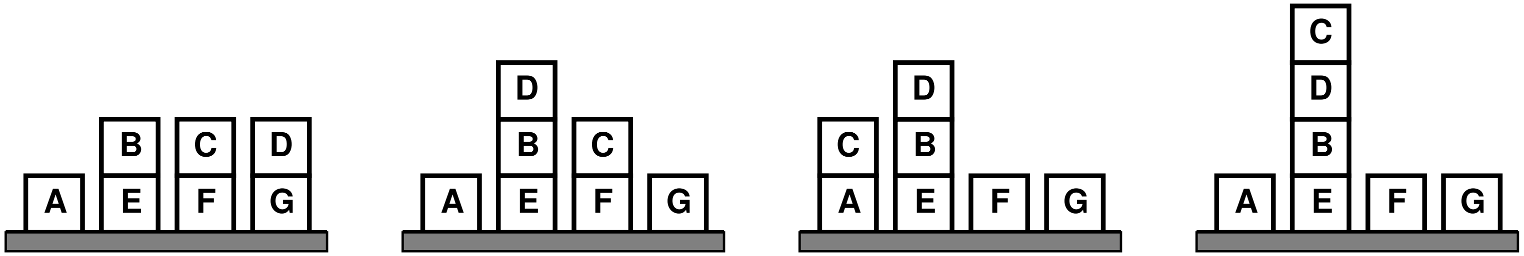
Goal state: $On(C, D) \wedge On(D, B)$

Action:

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

Move (x, y)

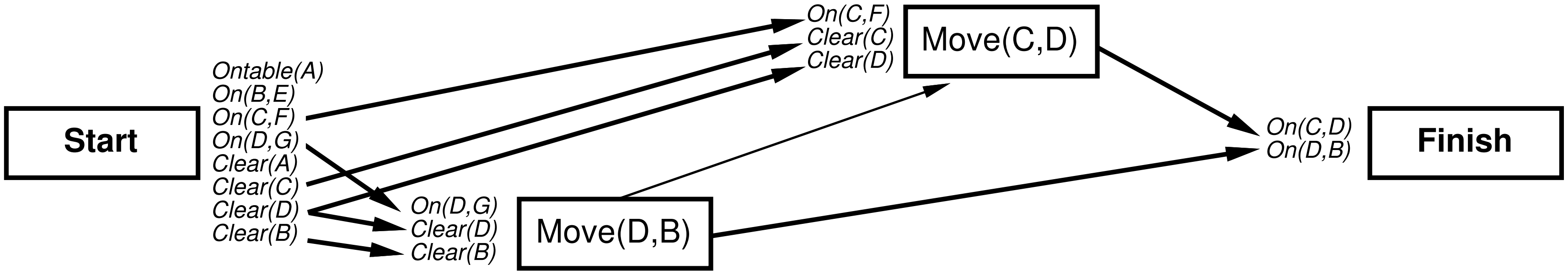
On(x, y), Clear(z), ~On(x,z), ~Clear(y)



(a) (b) (c) (d)

First, just like a POP agent builds a plan

Note: ordering constraints between both actions protects $Clear(D)$



Accident:

Someone moves D to B (illustration (b))

Current step description

$Clear(B)$, $On(D, G)$ no longer hold

Causal links become invalid and must be removed

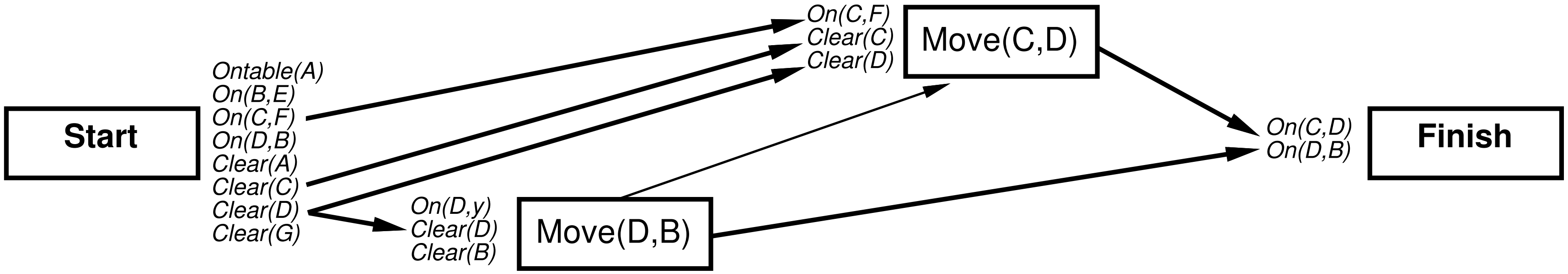
Preconditions for $Move(D, B)$ are now open

Precondition $On(D, y)$ is now uninstantiated

Causal link from $Move(D, B)$ to achieve $On(D, B)$ replaced by direct link to start node

→ extending a causal link

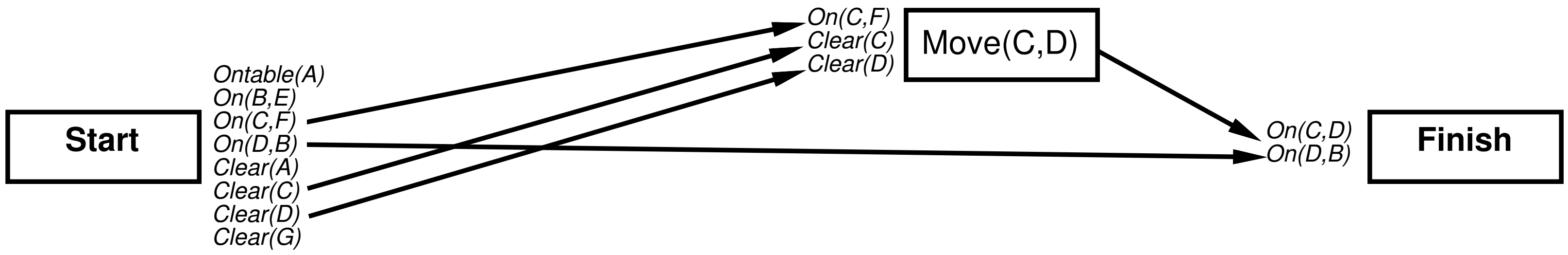
→ done every time a condition can be supplied by an earlier step without causing any threat



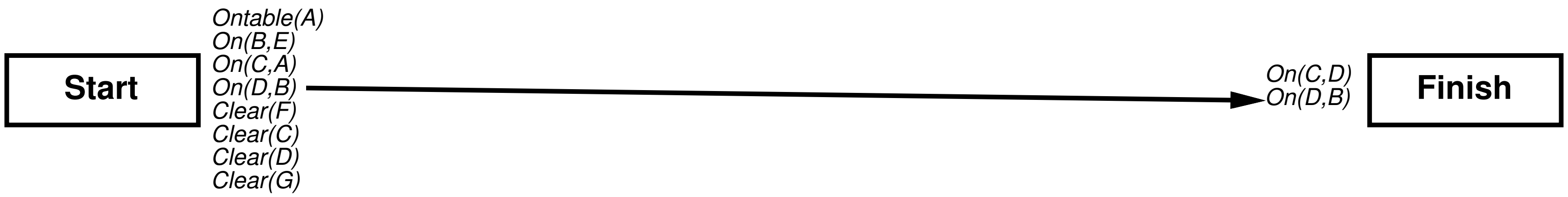
$Move(D, B)$ becomes a redundant step

→ thus, removed

Now, we can execute $Move(C, D)$
$\rightarrow$ all precond are satisfied, no threat



Accident: Agent drops C onto A instead of D

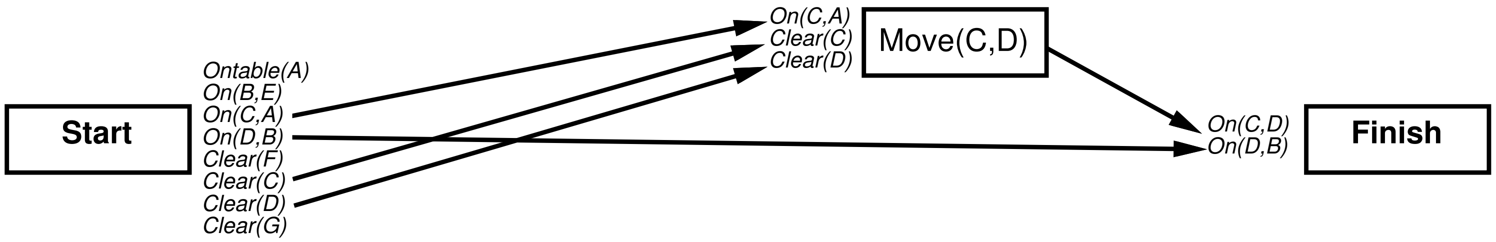


No action, but $On(C, D)$ still open..

Agent plans again as a simple POP

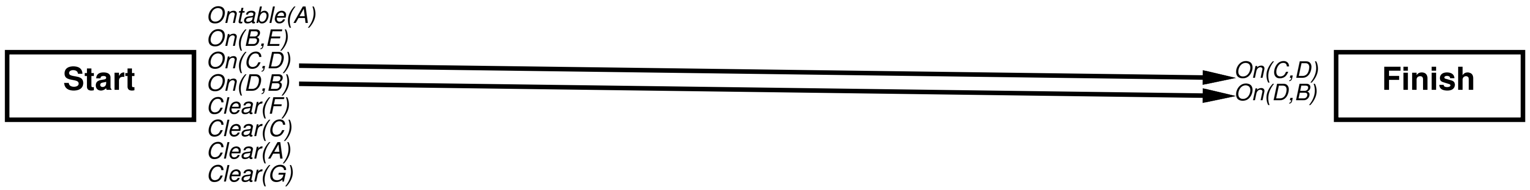
→ adds step $Move(C, D)$ to establish open condition

→ adds causal links from Start step



$Move(C, D)$ is executed, removed from plan

$On(C, D)$ of Finish is open again, linked to $On(C, D)$ of Start



Goal conditions are satisfied, no remaining actions.. Done!

```
function SITUATED-PLANNING-AGENT(percept) returns an action
  static: KB, a knowledge base (includes action descriptions)
          p, a plan, initially NoPlan
          t, a counter, initially 0, indicating time
          G, a goal

  TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
  current ← STATE-DESCRIPTION(KB, t)
  EFFECTS(START(p)) ← current
  if p = NoPlan then
     G ← ASK(KB, MAKE-GOAL-QUERY(t))
     p ← MAKE-PLAN(current, G, KB)
  action ← NoOp (the default)

   Termination:
     if there are no open preconditions and p has no steps other than START and FINISH then
        p ← NoPlan and skip remaining steps

   Resolving standard flaws:
     resolve any open condition by adding a causal link from any existing
        possibly prior step or a new step
     resolve potential threats by promotion or demotion

   Remove unsupported causal links:
     if there is a causal link START --c--> S protecting a proposition c
       that no longer holds in START then
        remove the link and any associated bindings

   Extend causal links back to earliest possible step:
     if there is a causal link S_j --c--> S_k such that
       another step S_i exists with S_i < S_j and the link S_i --c--> S_k is safe then
        replace S_j --c--> S_k with S_i --c--> S_k

   Remove redundant actions:
     remove any step S that supplies no causal links

   Execute actions when ready for execution:
     if a step S in the plan other than FINISH satisfies the following:
        (a) all preconditions satisfied by START;
        (b) no other steps necessarily between START and S; and
        (c) S does not threaten any causal link in p then
            add ordering constraints to force all other steps after S
            remove S from p, and all causal links to and from S
            action ← the action in S

  TELL(KB, MAKE-ACTION-SENTENCE(action, t))
  t ← t + 1
  return action
```

List of terms for glossary:

Due date: Monday, April 17, 2000

Words in margins in AIMA +

— clobbering (in planning)