Planning

CSE 473
Chapters 10.3 and 11

Planning

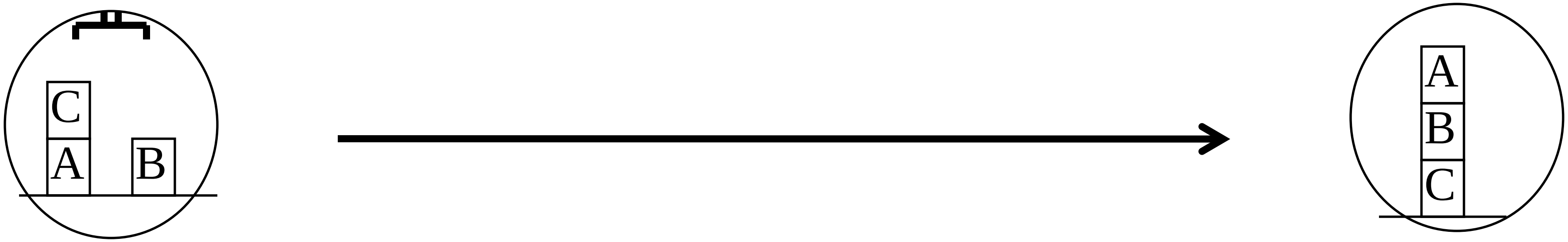
- Given

  a logical description of the **initial situation**,
  a logical description of the **goal conditions**, and
  a logical description of a set of **possible actions**,

- find

  a **sequence of actions** (a **plan of action**) that brings us from the initial situation to a situation in which the goal conditions hold.

Example: BlocksWorld



Planning Input: State Variables/Propositions

- Types: block --- a, b, c
- (on-table a) (on-table b) (on-table c)
- (clear a) (clear b) (clear c)
- (arm-empty)
- (holding a) (holding b) (holding c)
- (on a b) (on a c) (on b a) (on b c) (on c a) (on c b)

No. of state variables =16

- (on-table ?b); clear (?b)
- (arm-empty); holding (?b)
- (on ?b1 ?b2)

No. of states = 2^{16}

No. of reachable states = ?

Planning Input: Actions

- pickup a b, pickup a c, …
- place a b, place a c, …
- pickup-table a, pickup-table b, …
- place-table a, place-table b, …

- pickup ?b1 ?b2
- place ?b1 ?b2
- pickup-table ?b
- place-table ?b

Total: 6 + 6 + 3 + 3 = 18 “ground” actions

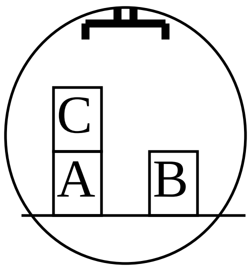
Total: 4 action schemata

Planning Input: Actions (contd)

- :action pickup ?b1 ?b2
  :precondition
  (on ?b1 ?b2)
  (clear ?b1)
  (arm-empty)
  :effect
  (holding ?b1)
  (not (on ?b1 ?b2))
  (clear ?b2)
  (not (arm-empty))

- :action pickup-table ?b
  :precondition
  (on-table ?b)
  (clear ?b)
  (arm-empty)
  :effect
  (holding ?b)
  (not (on-table ?b))
  (not (arm-empty))

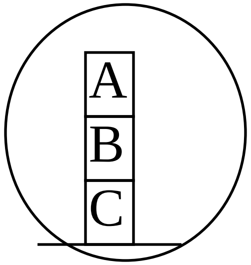
Planning Input: Initial State



- (on-table a) (on-table b)
- (arm-empty)
- (clear c) (clear b)
- (on c a)

- All other propositions false
  - not mentioned → false

Planning Input: Goal



- (on-table c) AND (on b c) AND (on a b)
- Is this a state?
- In planning a goal is a set of states

Planning Input Representation

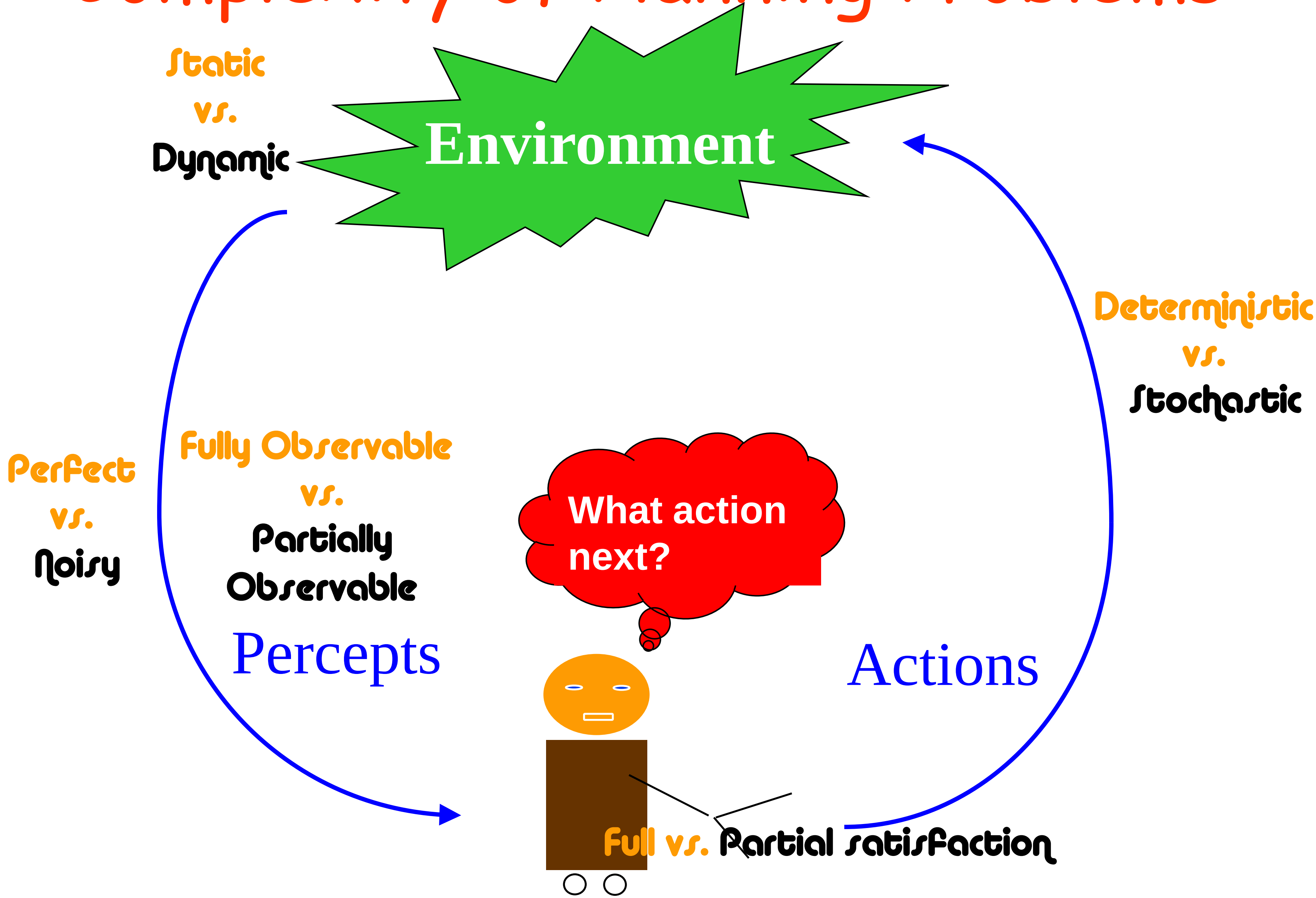
- Description of initial state of world
  - Set of propositions
- Description of goal: i.e. set of worlds
  - E.g., Logical conjunction
  - Any world satisfying conjunction is a goal
- Description of available actions

Planning vs. Problem-Solving

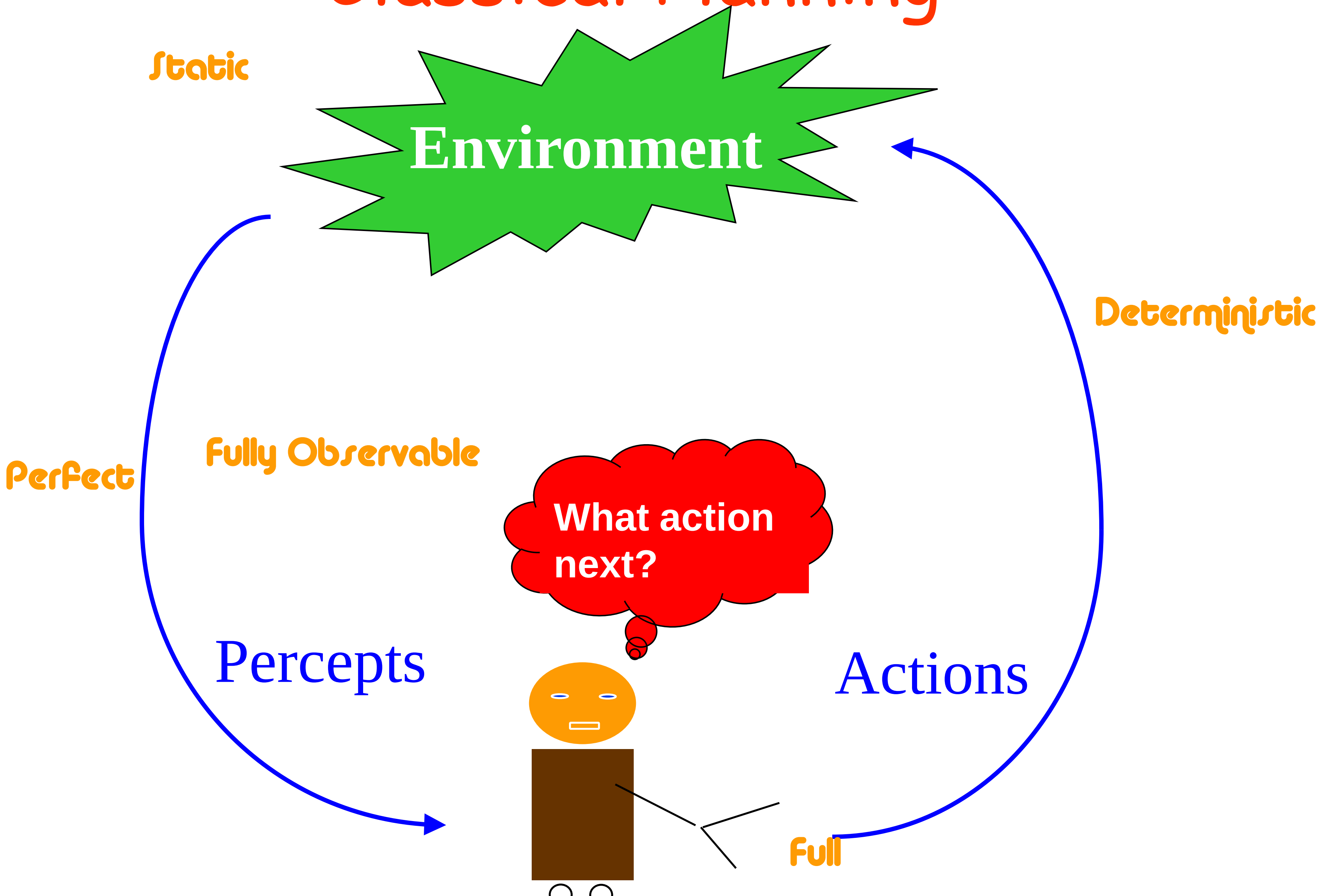
Basic difference: **Explicit, logic-based representation**

- States/Situations: descriptions of the world by logical formulae
→ agent can explicitly reason about and communicate with the world.
- Goal conditions as logical formulae vs. goal test (black box)
→ agent can reflect on its goals.
- Operators/Actions: Axioms or transformation on formulae in a logical form
→ agent can gain information about the effects of actions by inspecting the operators.

Complexity of Planning Problems



Classical Planning



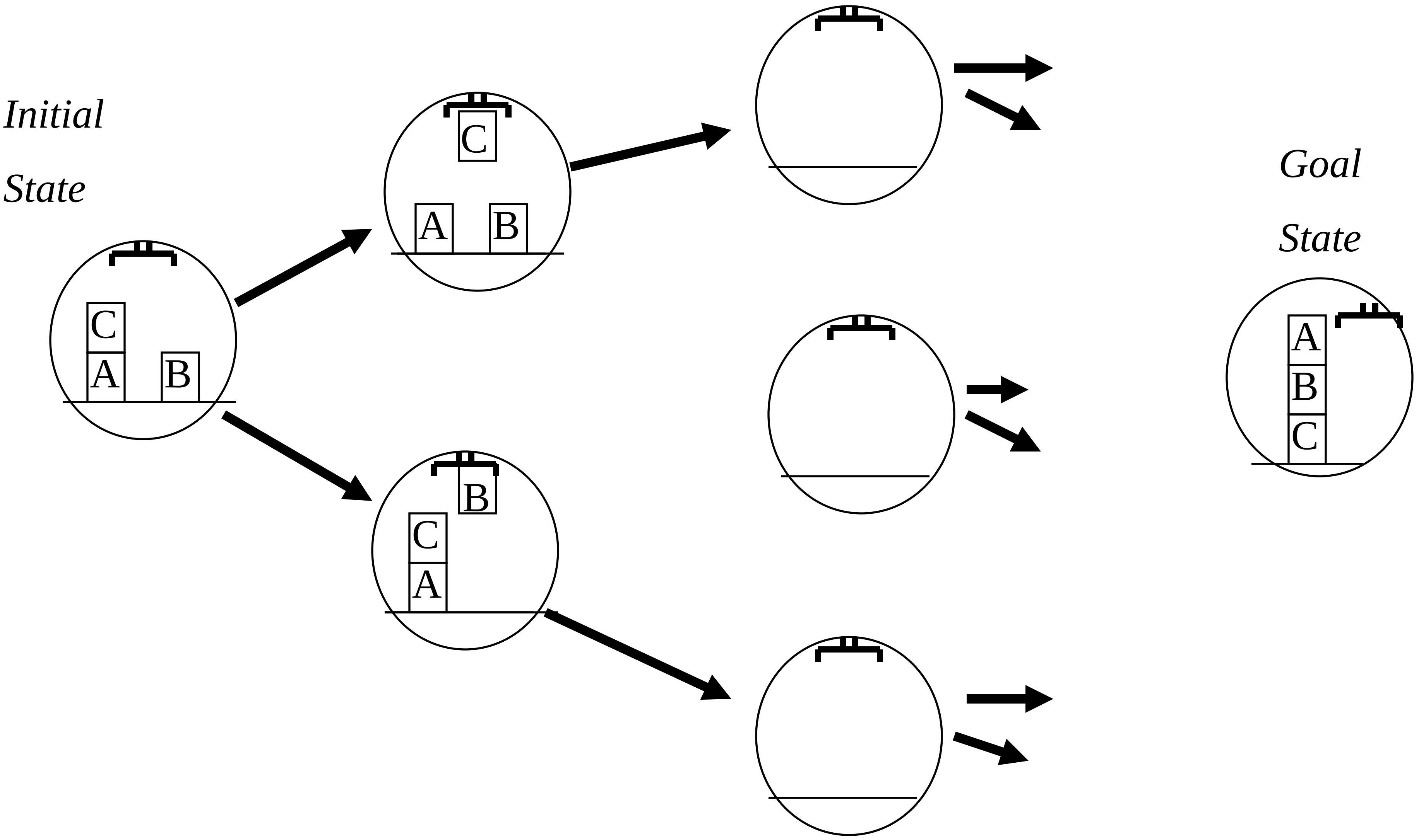
Actions in Classical Planning

- Simplifying assumptions
  - Atomic time
  - Agent is omniscient (no sensing necessary).
  - Agent is sole cause of change
  - Actions have deterministic effects
- STRIPS representation
  - World = set of true propositions (conjunction)
  - Actions:
    - Precondition: (conjunction of *positive* literals, no functions)
    - Effects (conjunction of literals, no functions)
  - Goal = conjunction of *positive* literals

  Is Blocks World in STRIPS?

  Goals = conjunctions (Rich ^ Famous)

Forward World-Space Search



Forward State-Space Search

- Initial state: set of positive ground literals (CWA: literals not appearing are false)
- Actions:
  - applicable if preconditions satisfied
  - add positive effect literals
  - remove negative effect literals
- Goal test: checks whether state satisfies goal
- Step cost: typically 1

Heuristics for State-Space Search

- Count number of false goal propositions in current state
  - Admissible?
  - NO

- Subgoal independence assumption:
  - Cost of solving conjunction is sum of cost of solving each subgoal independently
  - Optimistic: ignores negative interactions
  - Pessimistic: ignores redundancy

  - Admissible? No
  - Can you make this admissible?

Heuristics for State Space Search (contd)

- Delete all preconditions from actions, solve easy relaxed problem, use length
  - Admissible?
  - YES
- Delete negative effects from actions, solve easier relaxed problem, use length
  - Admissible?
  - YES (if Goal has only positive literals, true in STRIPS)

Complexity of Planning

- Size of Search Space

  size of world state space

- Size of World state space

  exponential in problem representation

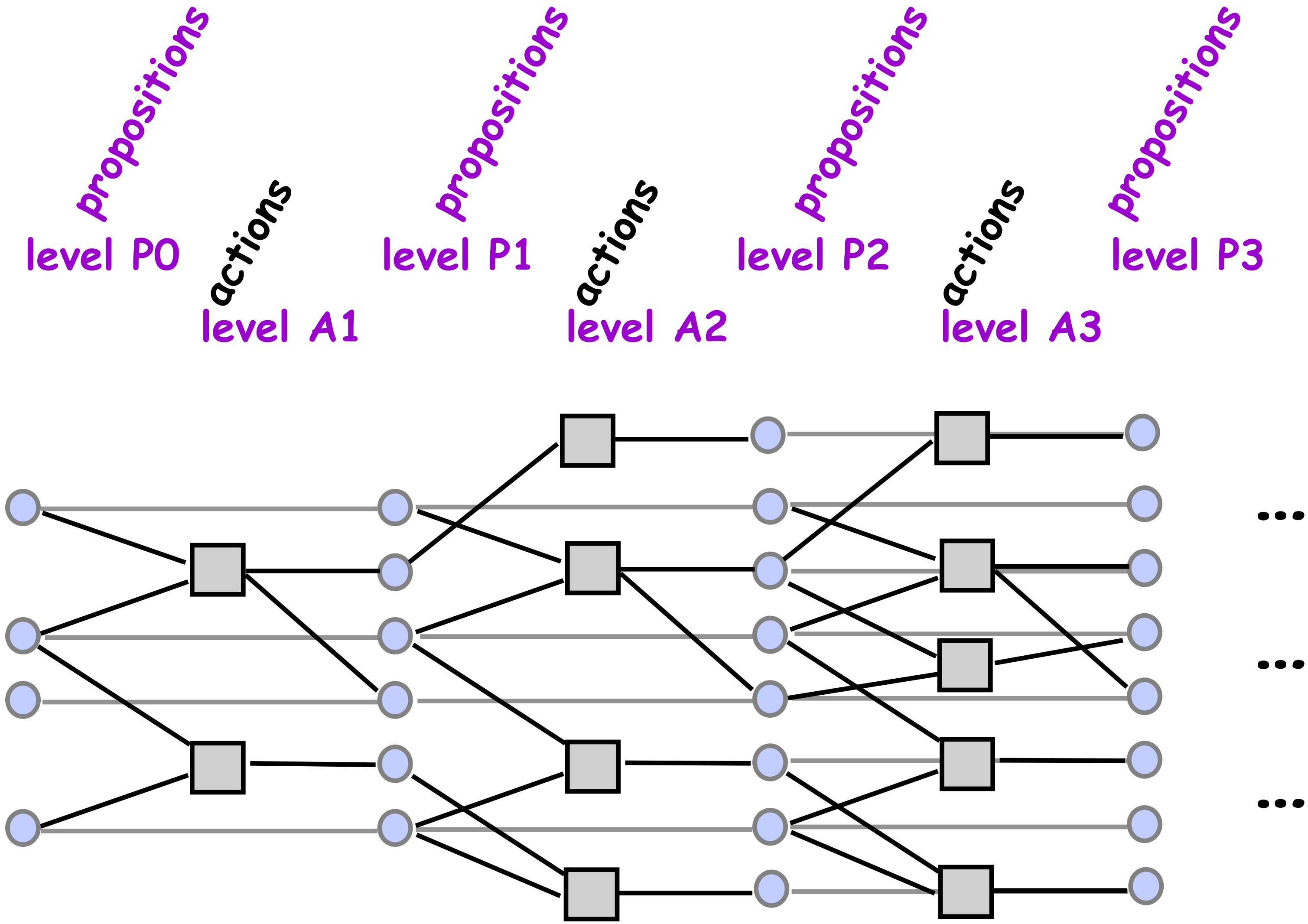
- What to do?

  Informative heuristic that can be computed in polynomial time!

Planning Graph: Basic idea

- Construct a planning graph: encodes constraints on possible plans
- Use this planning graph to constrain search for a valid plan (GraphPlan Algorithm):
  - If valid plan exists, it's a subgraph of the planning graph
- Use this planning graph to compute an informative heuristic (Forward A*)
- Planning graph can be built for each problem in polynomial time

The Planning Graph



Note: a few noops missing for clarity

Graph Expansion

Proposition level 0

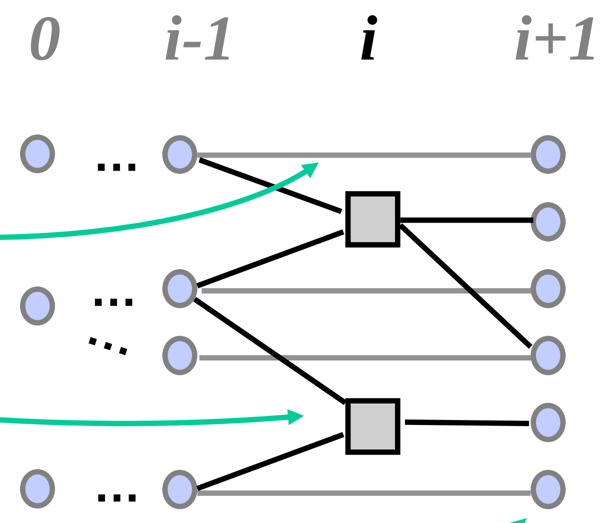
- initial conditions

Action level i

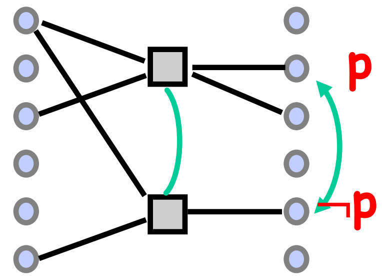
- no-op for each proposition at level i-1
- action for each operator instance whose
  preconditions exist at level i-1

Proposition level i

- effects of each no-op and action at level i

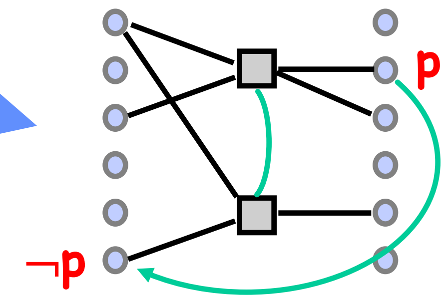


Mutual Exclusion



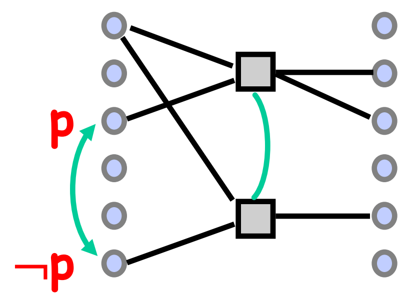
Two actions are mutex if

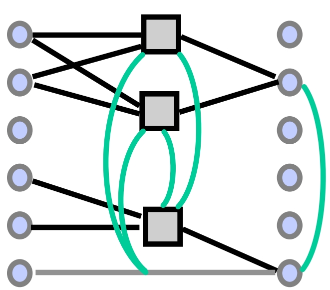
- one clobbers the other's effects or preconditions
- they have mutex preconditions



Two proposition are mutex if

- one is the negation of the other
- all ways of achieving them are mutex





Dinner Date

Initial Conditions: (:and (cleanHands) (quiet))

Goal: (:and (noGarbage) (dinner) (present))

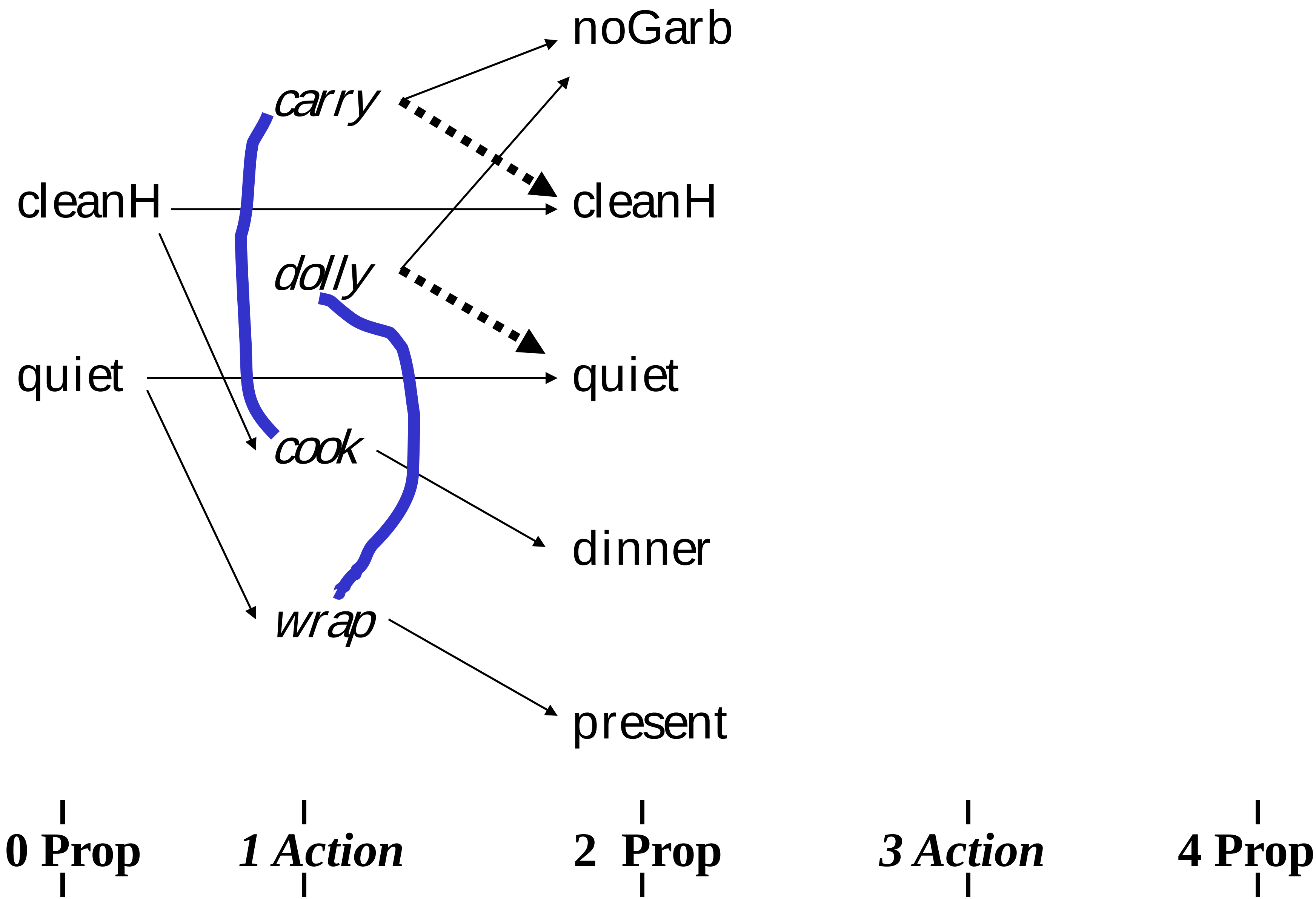
Actions:

(:operator **carry** *:precondition*
:effect (:and (noGarbage) (:not (cleanHands)))
(:operator **dolly** *:precondition*
:effect (:and (noGarbage) (:not (quiet)))
(:operator **cook** *:precondition* (cleanHands)
:effect (dinner))
(:operator **wrap** *:precondition* (quiet)
:effect (present))

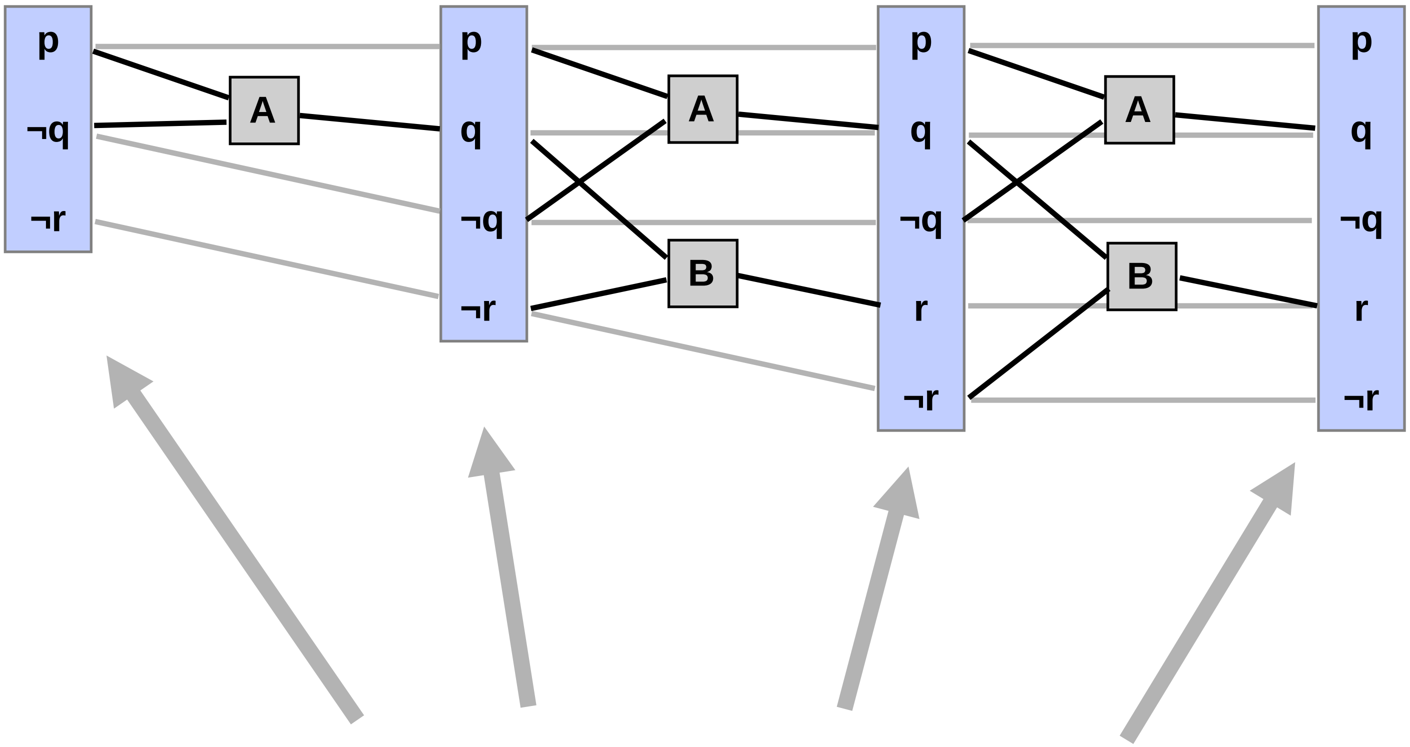
Planning Graph

0 Prop	1 Action	2 Prop	3 Action	4 Prop
		noGarb		
	carry			
cleanH		cleanH		
	dolly			
quiet		quiet		
	cook			
		dinner		
	wrap			
		present		

Are there any exclusions?



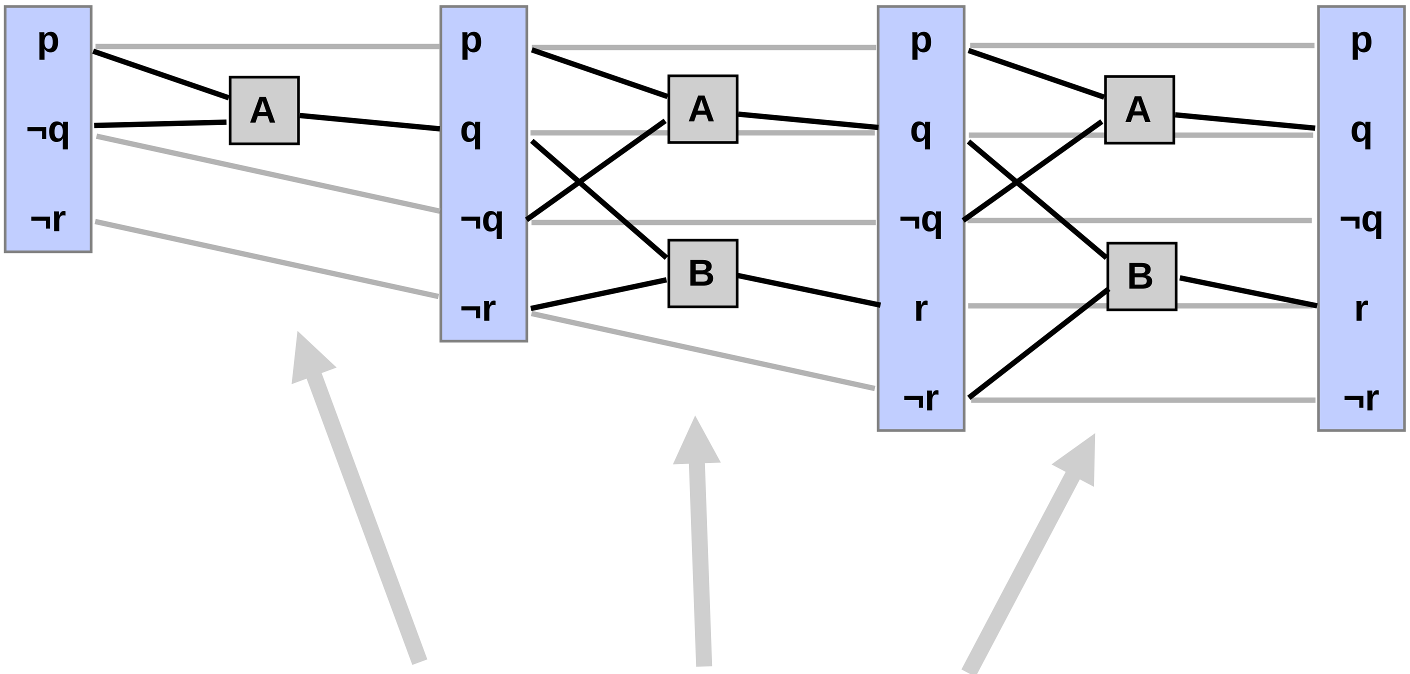
Observation 1



Propositions monotonically increase

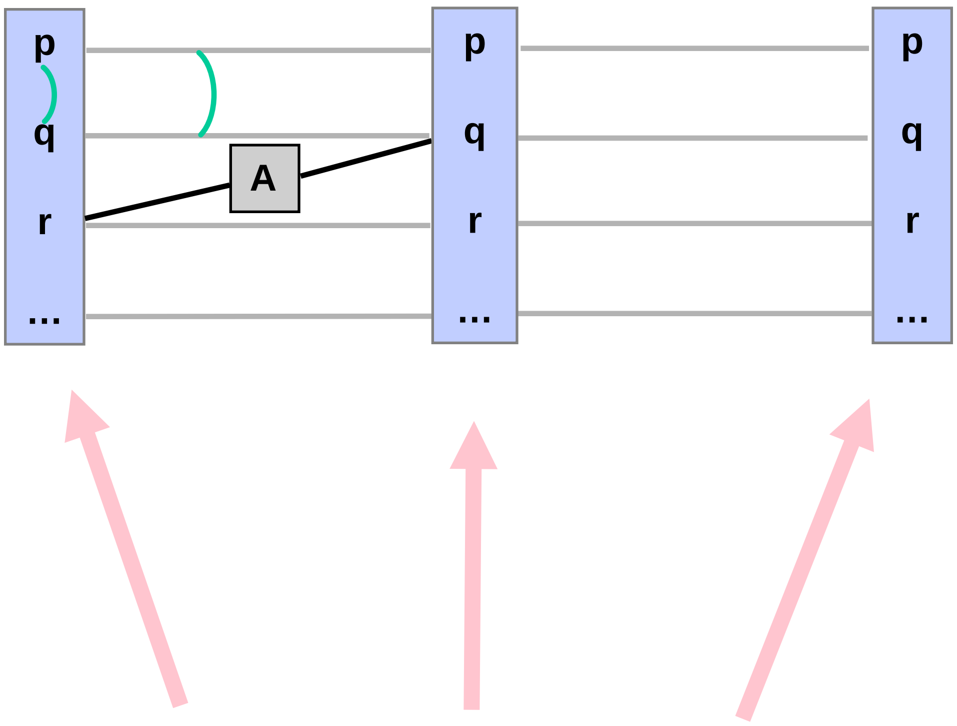
(always carried forward by no-ops)

Observation 2



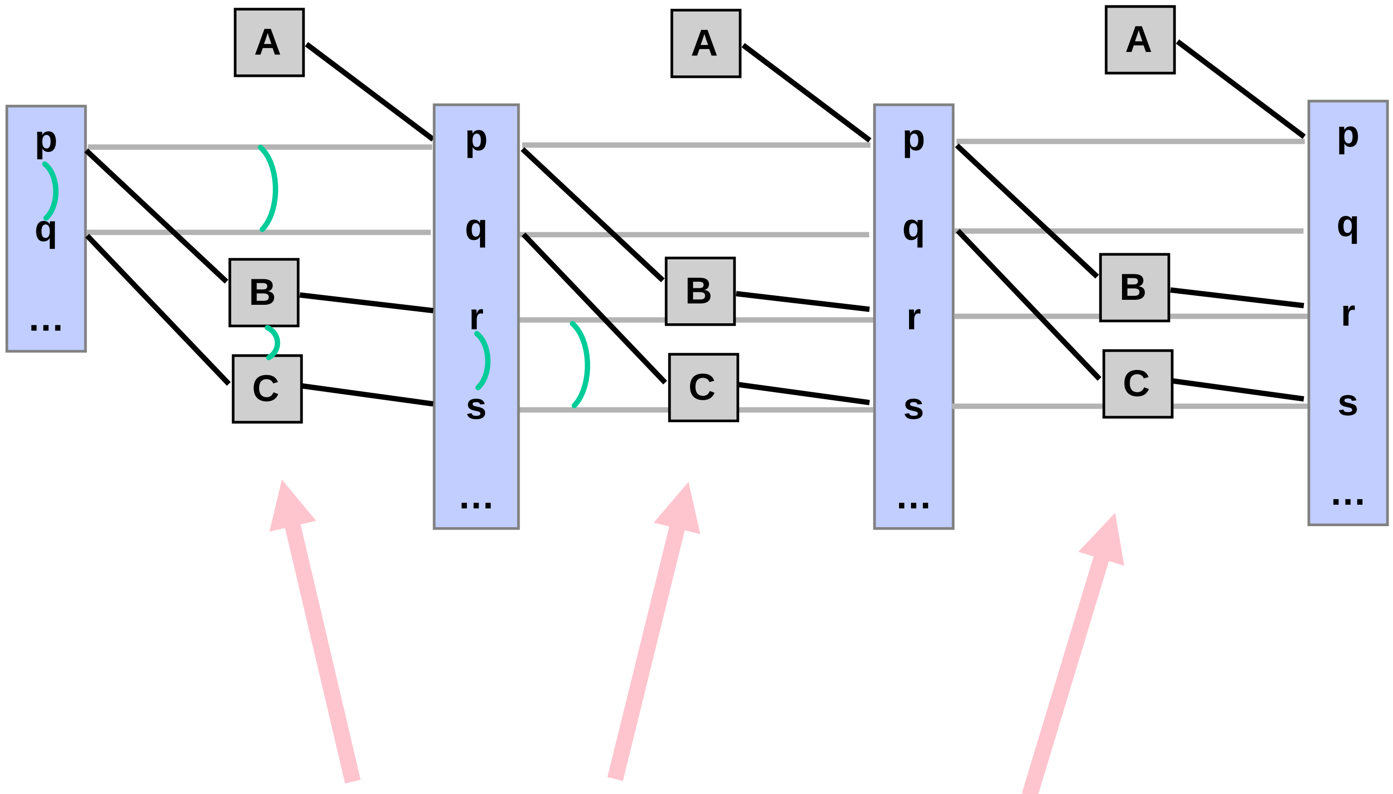
Actions monotonically increase

Observation 3



Proposition mutex relationships monotonically decrease

Observation 4



Action mutex relationships monotonically decrease

Observation 5

Planning Graph 'levels off'.

- After some time k all levels are identical
- Because it's a finite space, the set of literals never decreases and mutexes don't reappear.

Properties of Planning Graph

- If goal is absent from last level

    Goal cannot be achieved!
- If there exists a path to goal

    goal is present in the last level

- If goal is present in last level

    there may not exist any path still

    extend the planning graph further

Heuristics based on Planning Graph

- Construct planning graph starting from s
- h(s) = level at which goal appears non-mutex

    Admissible?

    YES

- Relaxed Planning Graph Heuristic

    Remove negative preconditions build plan. graph

    Use heuristic as above

    Admissible? YES

    More informative? NO

    Speed: FASTER