EECS 496: Sequential Decision Making

Soumya Ray

sray@case.edu

Office: Olin 516

Office hours : T 4-5:30 or by appointment

Recap

- Graphplan is a ___ ___ planner that combines: (i) ___________, (ii) ______________ and (iii) ____________.
- What is a layered plan?
- When is a layered plan valid?
- Is every POP a layered plan?
- Why is it useful to consider layered plans as the solution space?
- What is the connection between planning and reachability analysis?
- Each “state” of the planning graph represents a _____ of ___ _____. These are the results of _____ sequence of ___ of a fixed _____.
- The planning graph construction has alternate _____ and _____ layers. The first layer has _______________.
- To grow the action layer, we consider __________. Then we ______ to get the next layer.
- What are maintenance actions? Why are they important?
- The planning graph has two key advantages: (i) ______ and (ii) _______.

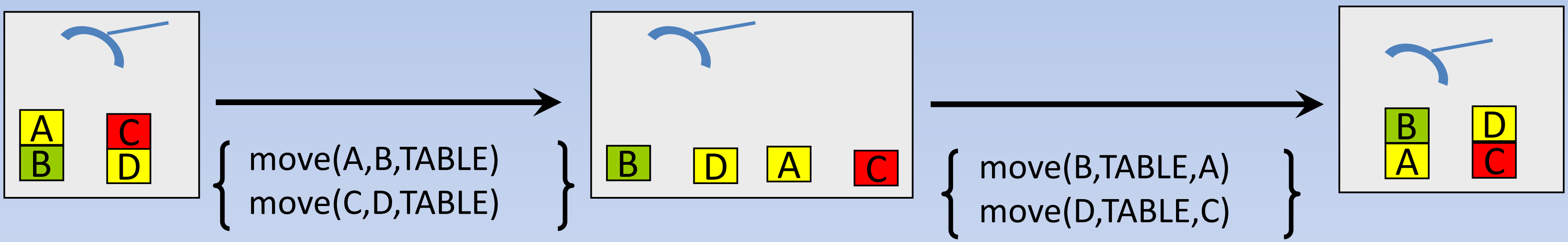
Today

- Graphplan

Graphplan (Blum and Furst 1994)

- A state-space planner that combines three key ideas:
  1. Layered Plans
  2. The Planning Graph
  3. Action Independence
- To create a fast general purpose planning algorithm
  – A basis for many state-of-the-art general purpose planners today

1. Layered Plan

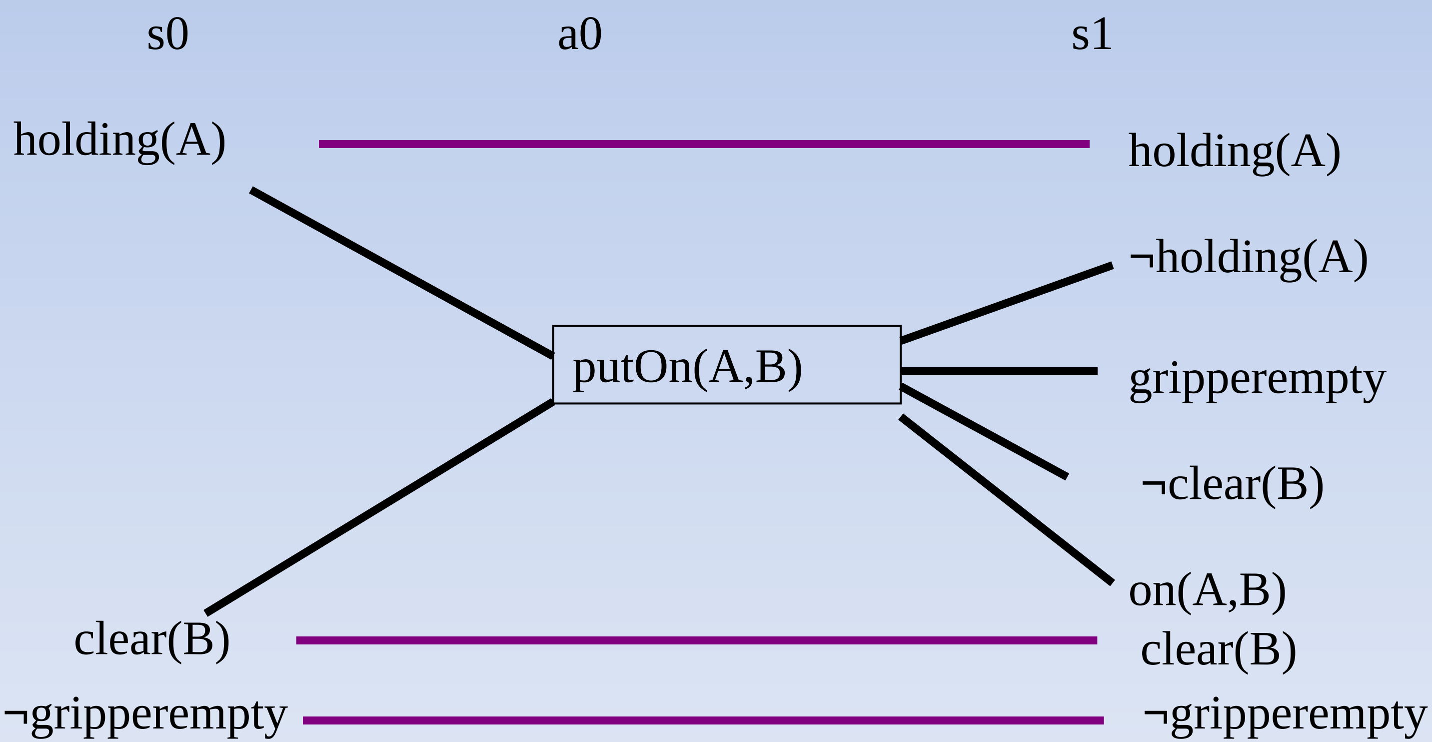


Planning Graph

putOn(A,B)

precondition: holding(A), clear(B)

effect: ¬holding(A), ¬clear(B), on(A,B), clear(A), gripperempty



3. Action Independence

- Key insight: By analyzing the action specifications, we can determine which actions can be executed at the same time and which can not

- This constrains the search for layered plans

3. Action Independence

- Two actions are independent iff the **delete** effects of one do not interfere with either the **preconditions** or the **add** effects of the other
  - Consequence: these actions can be done in any order and the same state will result

- Valid layered plans can only include independent actions in each layer

3. Dependent Actions

- If two actions a and b are not independent, then
  - a deletes an add effect of b : result state depends on order
  - a deletes a precondition of b : $b < a$
  - (vice versa for b)

3. Mutual Exclusion (“Mutex”) Relations

- Mutex relations track:
  - Propositions that can’t simultaneously be true at some level of the graph
  - Actions that can’t simultaneously be executed at some level of the graph

- Graphplan maintains all *pairwise* mutex relations as it works

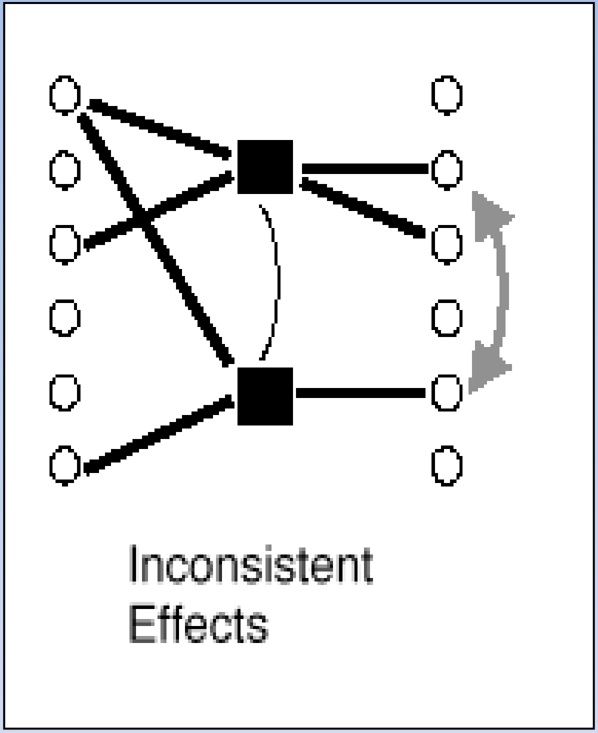
3. Mutex Relation 1: Actions

- Inconsistent effects
  - An effect of one negates an effect of the other

- E.g., putOn(A,B) & takeOff(A,B)

add gripperempty

delete gripperempty
(add ¬gripperempty)

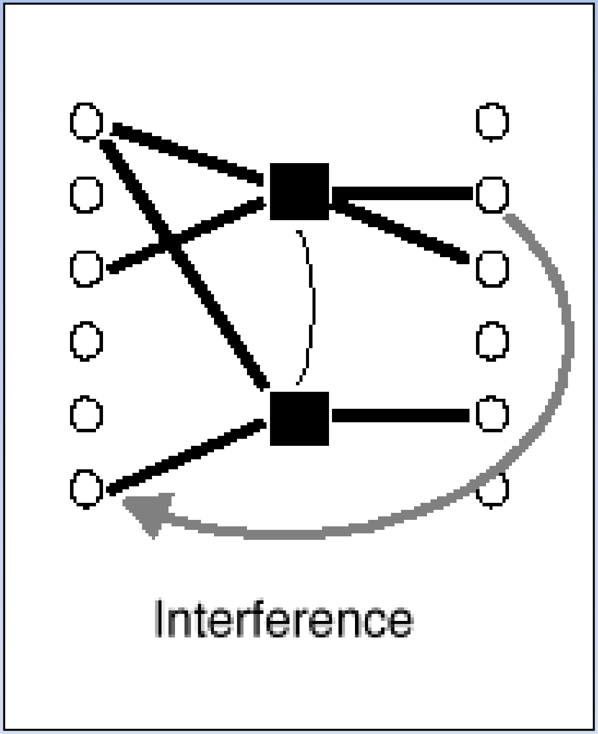


3. Mutex Relation 2: Actions

- Interference
  - An effect of one deletes a precondition of the other

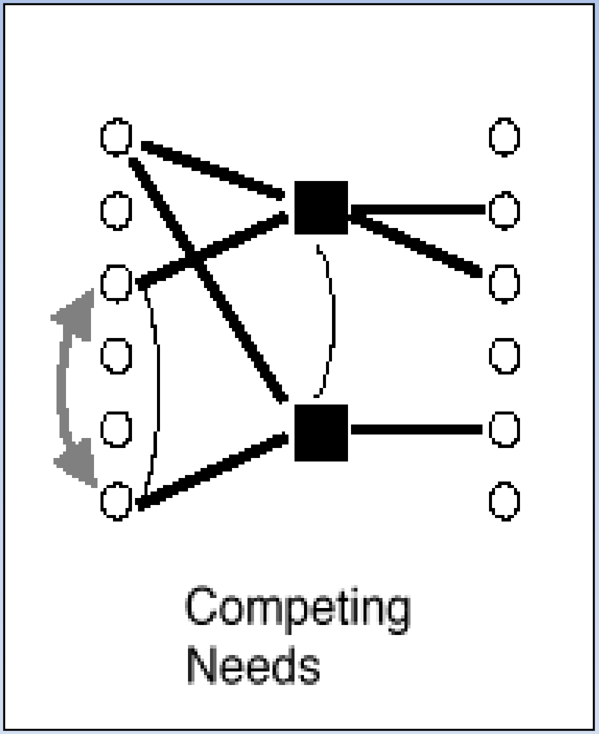
- E.g., putOn(A,B) & putDown(A)

deletes holding(A) needs holding(A)



3. Mutex Relation 3: Actions

- Competing needs
  - They have mutually exclusive preconditions



3. Mutex Relations: Actions

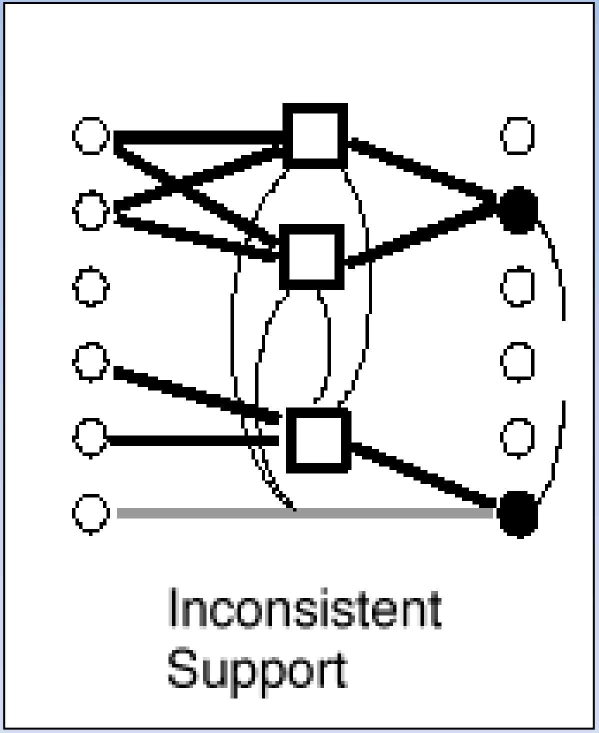
- Two actions a and b are mutex at level i of the plan graph if
  - a and b are dependent, **or**
  - A precondition of a is mutex with a precondition of b

3. Mutex Relations: Propositions

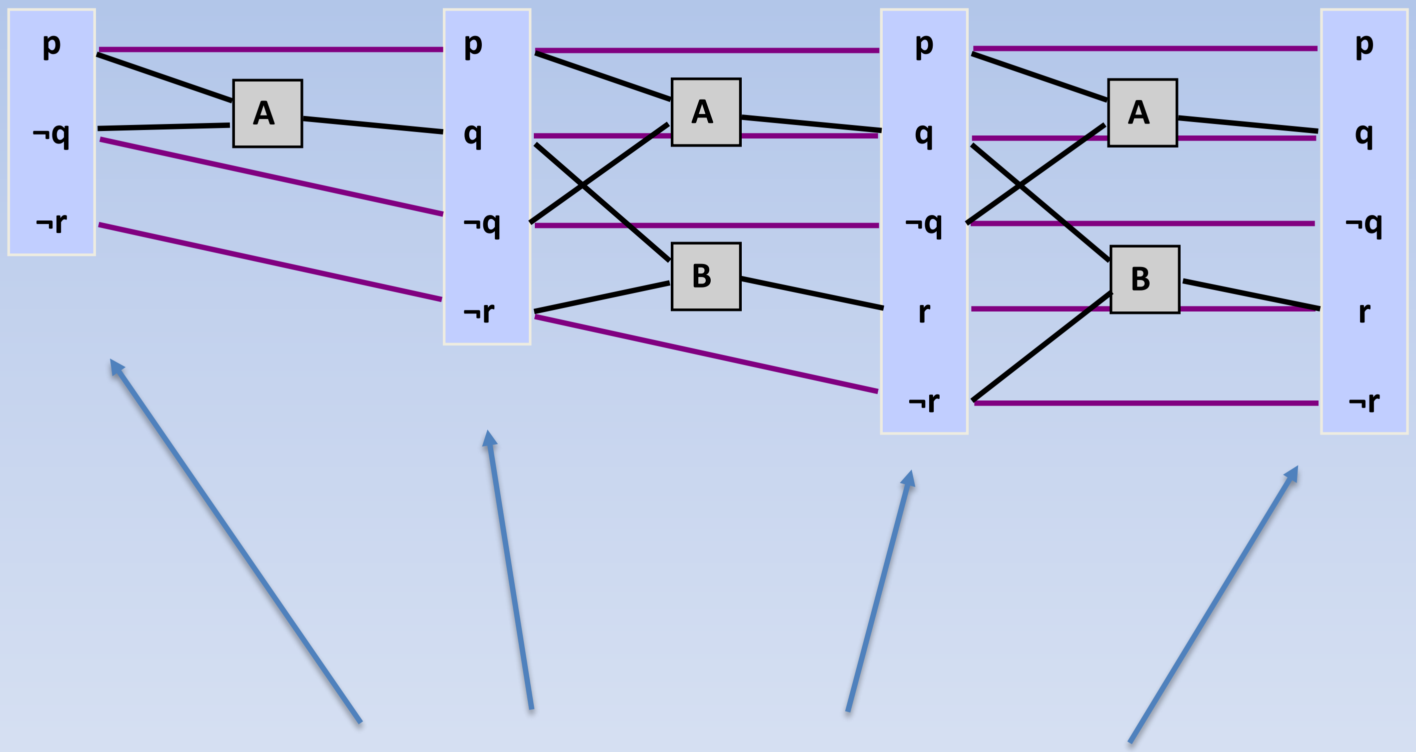
- One is the negation of the other
  E.g., gripperempty and ¬gripperempty

3. Mutex Relations: Propositions

- Two propositions p and q ($\neq \neg p$) have ***inconsistent support*** at level i of the plan graph if:
  - There is no action at level i that produces both p and q **and**
  - Every action that produces p is mutex with every action that produces q

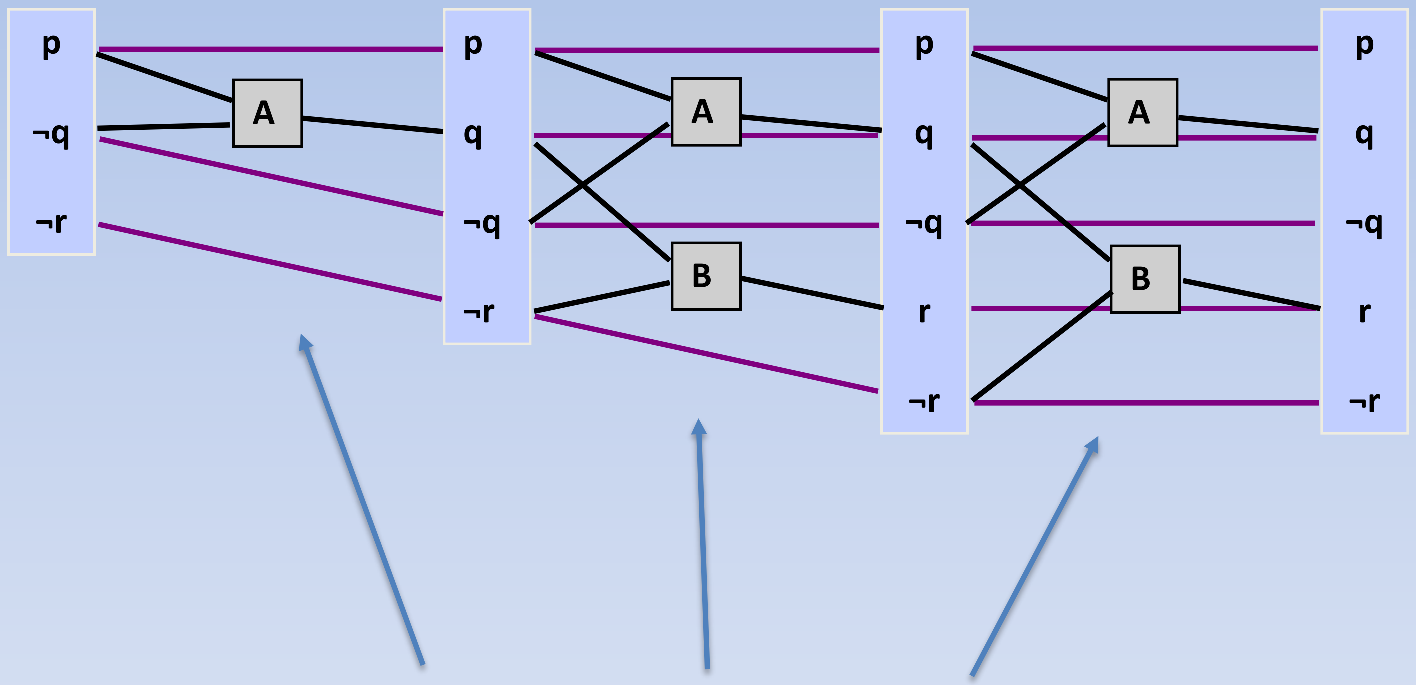


Key properties of Planning Graph: 1



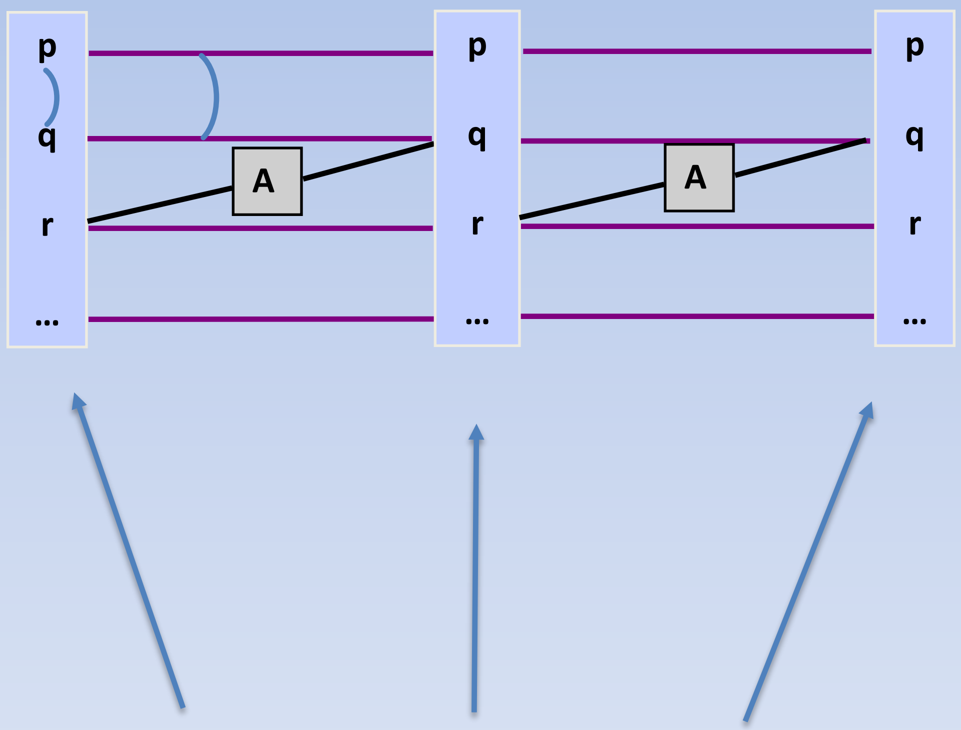
Propositions monotonically increase
(always carried forward by maintenance actions)

Key properties of Planning Graph: 2



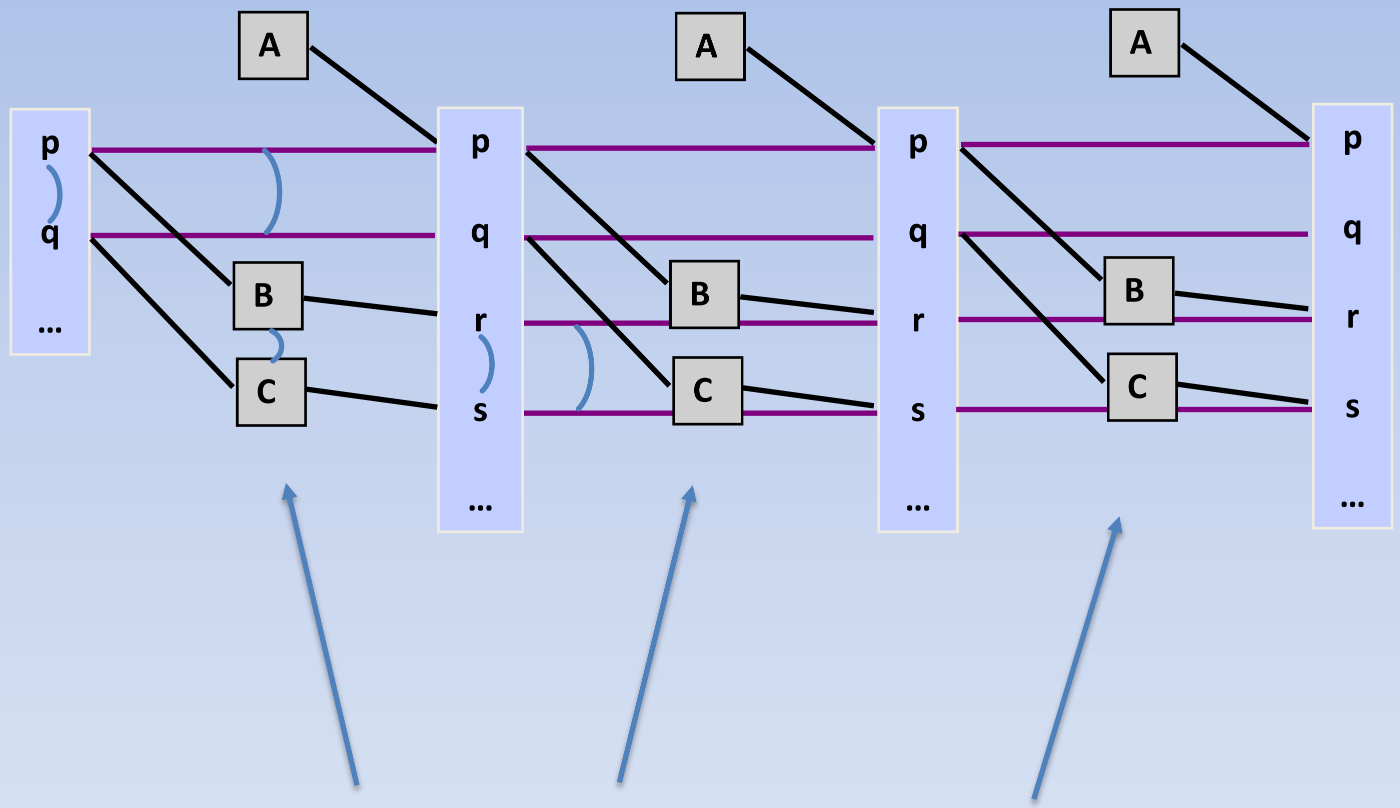
Actions monotonically increase

Key properties of Planning Graph: 3



- **Proposition mutex relationships monotonically decrease**
- **Specifically, if *p* and *q* are in layer *n* and are not mutex then they will not be mutex in future layers.**

Key properties of Planning Graph: 4



Action mutex relationships monotonically decrease

Key properties of Planning Graph: 5

Planning Graph "levels off"

- After some time *k* all levels are identical
- This is because there are a finite number of propositions and actions, the set of propositions never decreases and mutexes don't reappear.

Key Properties of Planning Graph: 6

- A goal g is reachable from the initial state only if all propositions in g appear in some level of the plan graph and none of them are mutex (*)
  - Necessary, not sufficient

Key Properties of Planning Graph: 7

- The size of the planning graph is polynomial in the size of the planning problem (see book)

Graphplan Algorithm

- Initialize the planning graph with initial state
- While not done
  - *Expand* the planning graph by one level
  - If new level is identical to old level (including mutexes), FAIL
  - If new level satisfies (*), check for solution
  - If found, stop
  - Else go to step 1

Expanding a Planning Graph

- Add an action layer
  - If all preconditions are in the previous layer and all nonmutex, add the action
- Add proposition layer
  - Add all effects of all actions in the action layer (including maintenance actions)
- Add action mutexes for new layer
- Add proposition mutexes for new layer

Solution Extraction

- If goals present and non-mutex
  - Choose any actions that satisfy the goals
  - Add actions' preconditions to new goals
  - Repeat until initial state is reached, or some level is reached where mutex relations hold

Example – Dinner Date

due to Dan Weld (U. of Washington)

- You want to prepare a surprise for your friend, who is asleep. The house is quiet and you’ve washed your hands. You need to make dinner, wrap a present and take out the garbage.

Example – Dinner Date

- Initial State: {cleanHands, quiet}
- Goal: {dinner, present, noGarbage}

Action	Preconditions	Effects
cook	cleanHands	dinner
wrap	quiet	present
carry	*none*	noGarbage, ¬cleanHands
dolly	*none*	noGarbage, ¬quiet