Planning: Forward Planning and CSP Planning

Alan Mackworth

UBC CS 322 – Planning 2

February 15, 2013

Textbook §8.2, 8.4

Reminders

- Reminders:
  - Assignment 2 was due today, 1pm
  - Midterm Wednesday, Mar 6: DMP 110, 3-3:50pm
    - ~60% short answer questions. See Connect now for full set.
    - ~40% long answer question.
    - Closed book – **non-programmable** calculator allowed.
    - See Connect now for previous midterm with solutions.
  - Giuseppe Carenini will lecture the week of Feb 25 – Mar 1.
  - Assignment 3 is out now. Due Wednesday, Mar 13, 1pm. Get started on Questions 1 & 2 now.
  - Exercises 6, 7, 8 and 9 posted. Do them.

Lecture Overview

- Recap: STRIPS and forward planning
- Heuristics for forward planning
- Planning as CSP
  - CSP representation
  - Solving the planning problem as CSP

Course Overview

Course Module

Representation

Reasoning Technique

Environment

Problem Type

		Deterministic	Stochastic	
Static	Constraint Satisfaction	*Variables + Constraints* Arc Consistency Search		
Static	Logic	*Logics* Search	*Bayesian Networks* Variable Elimination	Uncertainty
Sequential	Planning	*STRIPS* Search As CSP (using arc consistency)	*Decision Networks* Variable Elimination *Markov Processes* Value Iteration	Decision Theory

Now we start planning

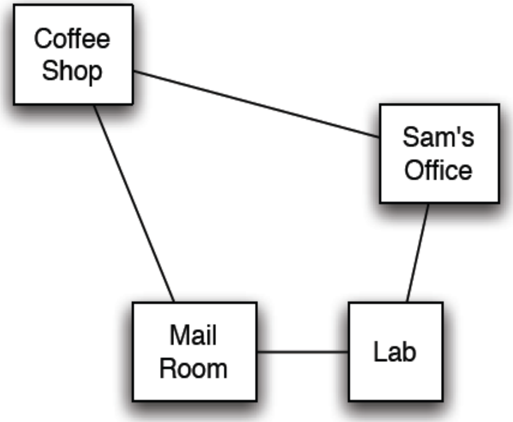
Key Idea of Planning

- Open up the representation of states, goals and actions
  - States and goals as features (variable assignments), as in CSP
  - Actions as preconditions and effects defined on features

- Agent can reason more deliberatively about what actions to consider to achieve its goals, rather than just using blind or heuristic search alone.

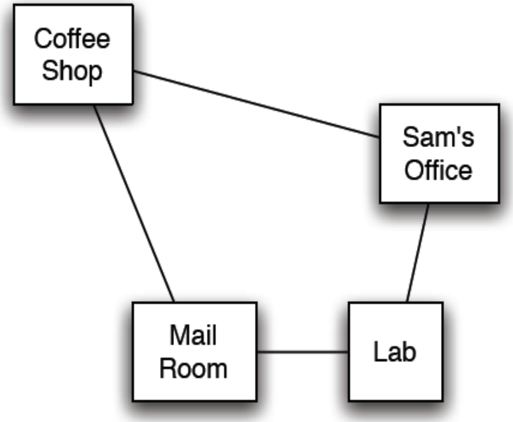
Delivery Robot Example: features

- RLoc - Rob's location
  - Domain: {coffee shop, Sam's office, mail room, laboratory} short {cs, off, mr, lab}
- RHC – Rob has coffee
  - Domain: {true, false}. By rhc indicate that Rob has coffee, and by $\overline{rhc}$ that Rob doesn't have coffee
- SWC – Sam wants coffee {true, false}
- MW – Mail is waiting {true, false}
- RHM – Rob has mail {true, false}



- An example state is $\langle lab, \overline{rhc}, swc, \overline{mw}, rhm \rangle$

Delivery Robot Example: Actions



The robot's actions are:

Move - Rob's move action

- move clockwise (mc), move anti-clockwise (mac)

PUC - Rob picks up coffee

- must be at the coffee shop

DelC - Rob delivers coffee

- must be at the office, and must have coffee

PUM - Rob picks up mail

- must be in the mail room, and mail must be waiting

DelM - Rob delivers mail

- must be at the office and have mail

Preconditions for action application

Example State-Based Representation

State	Action	Resulting State
$\langle lab, \overline{rhc}, swc, \overline{mw}, rhm \rangle$	$\langle mc \rangle$	$\langle mr, \overline{rhc}, swc, \overline{mw}, rhm \rangle$
$\langle lab, \overline{rhc}, swc, \overline{mw}, rhm \rangle$	$\langle mac \rangle$	$\langle off, \overline{rhc}, swc, \overline{mw}, rhm \rangle$
$\langle off, \overline{rhc}, swc, \overline{mw}, rhm \rangle$	$\langle dm \rangle$	$\langle off, \overline{rhc}, \overline{swc}, \overline{mw}, \overline{rhm} \rangle$
$\vdots$	$\vdots$	$\vdots$

Tabular representation:

need an entry for every state and every action applicable in that state!

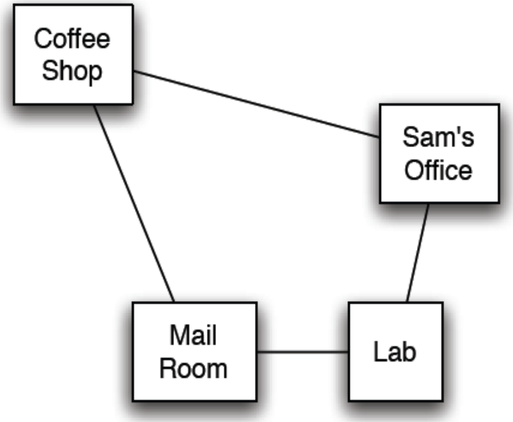
STRIPS representation

In STRIPS, an action has two parts:

1. Preconditions: a set of assignments to variables that must be satisfied in order for the action to be legal

2. Effects: a set of assignments to variables that are caused by the action

STRIPS example



- In STRIPS, an action has two parts:
  - 1. Preconditions: a set of assignments to variables that must be satisfied in order for the action to be legal
  - 2. Effects: a set of assignments to variables that are caused by the action

- STRIPS representation of the action pick up coffee, PUC:
  - preconditions Loc = cs and RHC = $\overline{rhc}$
  - effects RHC = rhc

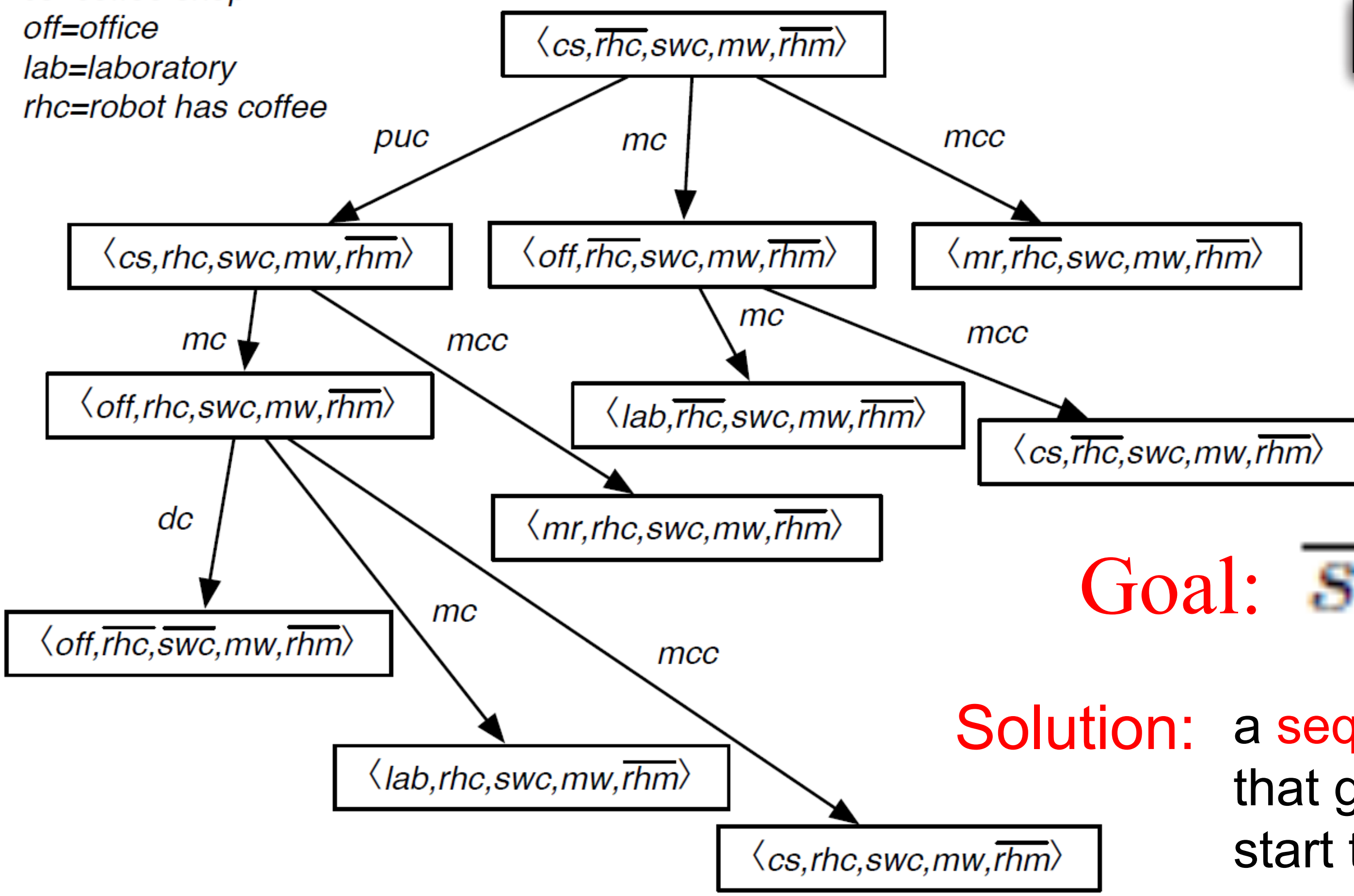
- STRIPS representation of the action deliver coffee, DelC:
  - preconditions Loc = off and RHC = rhc
  - effects RHC = $\overline{rhc}$ and SWC = $\overline{swc}$

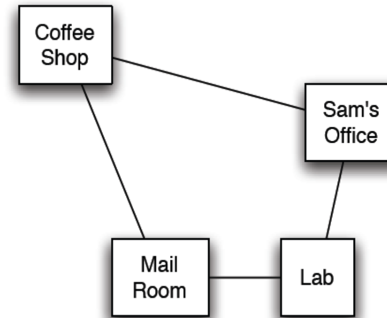
Standard Search vs. Specific R&R systems

- Constraint Satisfaction (Problems):
  - State: assignments of values to a subset of the variables
  - Successor function: assign values to a "free" variable
  - Goal test: set of constraints
  - Solution: possible world that satisfies the constraints
  - Heuristic function: none (all solutions at the same distance from start)
- Planning :
  - State: full assignment of values to features
  - Successor function: states reachable by applying valid actions
  - Goal test: partial assignment of values to features
  - Solution: a sequence of actions
  - Heuristic function: next time
- Inference
  - State
  - Successor function
  - Goal test
  - Solution
  - Heuristic function

Example for state space graph

cs=coffee shop
off=office
lab=laboratory
rhc=robot has coffee





Goal: $\overline{swc}$

Solution: a sequence of actions that gets us from the start to a goal

What is a solution to this planning problem?

(puc, mc)

(puc, mc, mc)

(puc, dc)

(puc, mc, dc)

Standard Search vs. Specific R&R systems

- Constraint Satisfaction (Problems):
  - State: assignments of values to a subset of the variables
  - Successor function: assign values to a “free” variable
  - Goal test: set of constraints
  - Solution: possible world that satisfies the constraints
  - Heuristic function: none (all solutions at the same distance from start)
- Planning :
  - State: full assignment of values to features
  - Successor function: states reachable by applying valid actions
  - Goal test: partial assignment of values to features
  - Solution: a sequence of actions
  - Heuristic function: now
- Inference
  - State
  - Successor function
  - Goal test
  - Solution
  - Heuristic function

Lecture Overview

- Recap: STRIPS and forward planning
- Heuristics for forward planning
- Planning as CSP
  - CSP representation
  - Solving the planning problem as CSP

Heuristics for Forward Planning

- Not in textbook, but you can see details in Russell & Norvig, 10.3.2
- Heuristic function: estimate of the distance from a state to the goal
- Good heuristics make forward planning feasible in practice
- In planning, the distance from a state s to the goal is

goal features not true in s

actions needed to get from s to the goal

legal actions in s

- Factored representation of states and actions allows for definition of domain-independent heuristics
  - Will see two examples: general heuristics, independent of domain

Heuristics for Forward Planning

- Recall general method for creating admissible heuristics
  - Relax the original problem

- One example: ignore preconditions; makes problem trivial

- Another example: ignore delete lists
  - Assumptions for simplicity:
    - All features are binary: T / F
    - Goals and preconditions can only be assignments to T
  - Every action has add list and delete list
    - Add list: features that are made true by the action
    - Delete list: features that are made false by the action
  - Compute heuristic values: solve relaxed problem without delete lists!
    - Planning is PSPACE-hard (that's **really** hard, includes NP-hard)
    - Without delete lists: often very fast
- These heuristics are covered in Assignment 3.

Lecture Overview

- Recap: STRIPS and forward planning
- Heuristics for forward planning
- Planning as CSP
  - CSP representation
  - Solving the planning problem as CSP

Planning as a CSP

- An alternative approach to planning is to set up a planning problem as a CSP

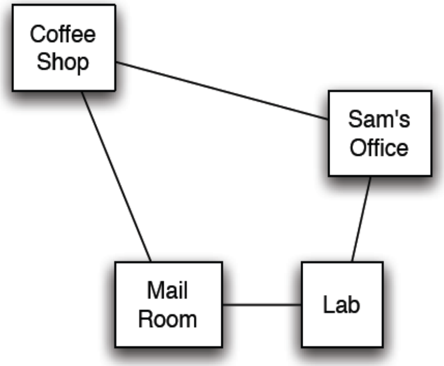
- We simply reformulate a STRIPS model as a set of variables and constraints

Planning as a CSP

- We simply reformulate a STRIPS model as a set of variables and constraints
- Give it a try: please work in groups of two or three for a few minutes and try to define what would you chose as
  - Variables
  - Constraints
- Use the Robot Delivery World as a leading example

What will be the CSP variables and constraints?

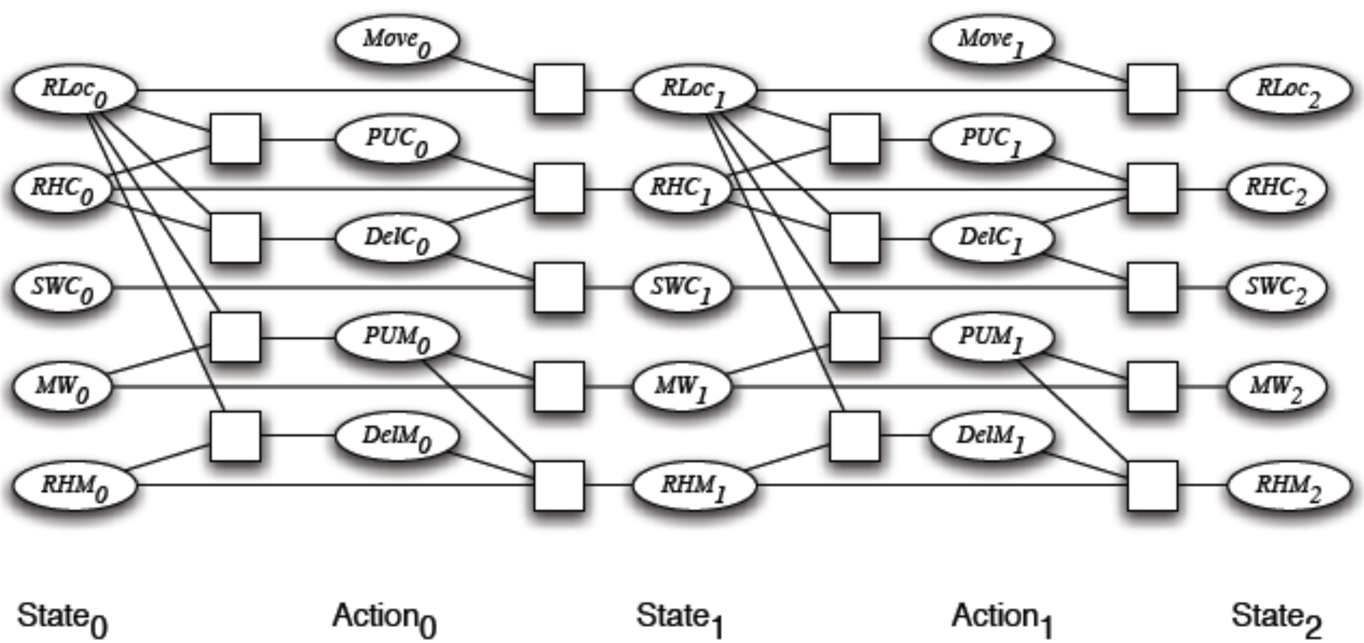
- Features change over time
  - Might need more than one CSP variable per feature
- Initial state constraints
- Goal state constraints



- STRIPS example actions
  - STRIPS representation of the action pick up coffee, PUC:
    - preconditions Loc = cs and RHC = $\overline{rhc}$
    - effects RHC = rhc
  - STRIPS representation of the action deliver coffee, DelC:
    - preconditions Loc = off and RHC = rhc
    - effects RHC = $\overline{rhc}$ and SWC = $\overline{swc}$
  - Have to capture these conditions as constraints

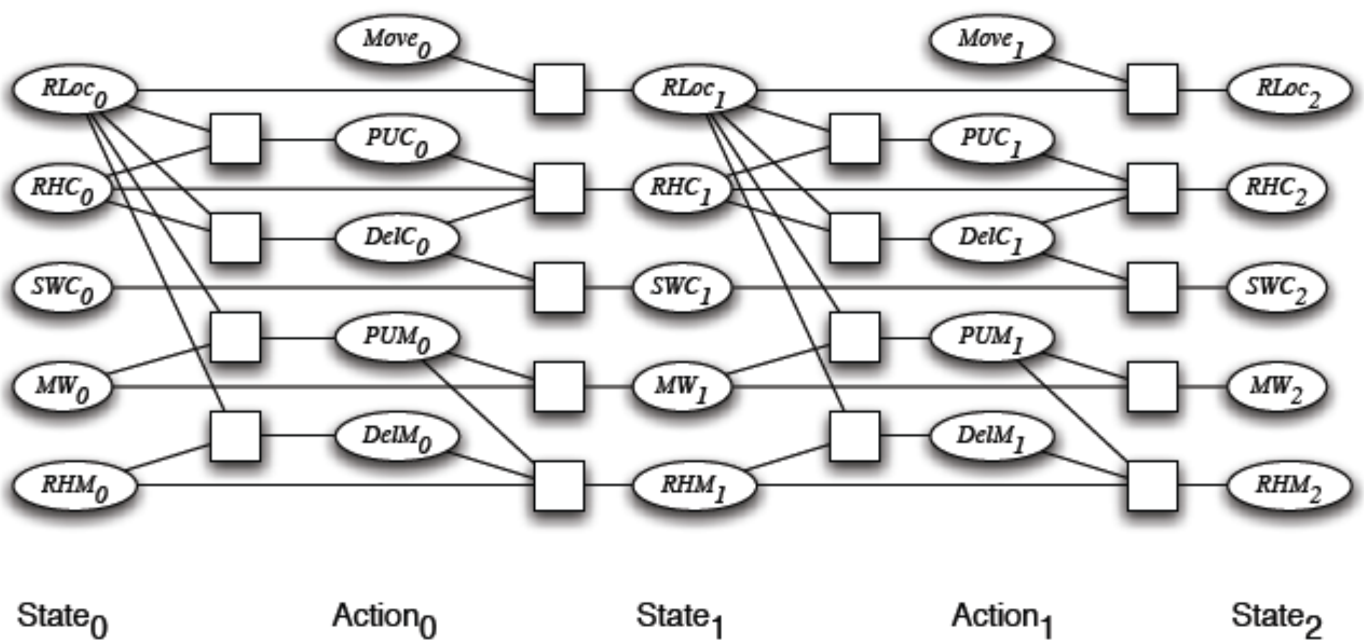
Planning as a CSP: General Idea

- Both features and actions are CSP variables
  - one CSP variable for each time step for each action and each feature
- Action preconditions and effects are constraints among
  – the action,
  – the states in which it can be applied
  – the states that it can generate



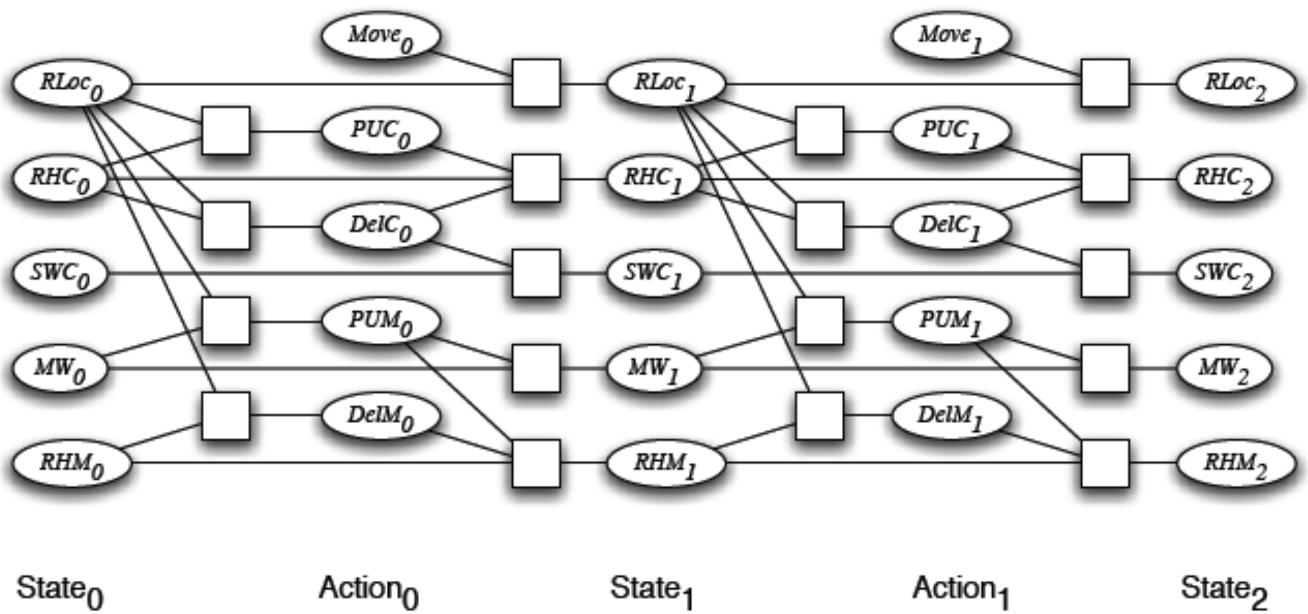
Planning as a CSP: General Idea

- These action constraints relate to states at a given time t, the corresponding valid actions and the resulting states at t +1
  - we need to have as many state and action variables as we have planning steps



Planning as a CSP: Variables

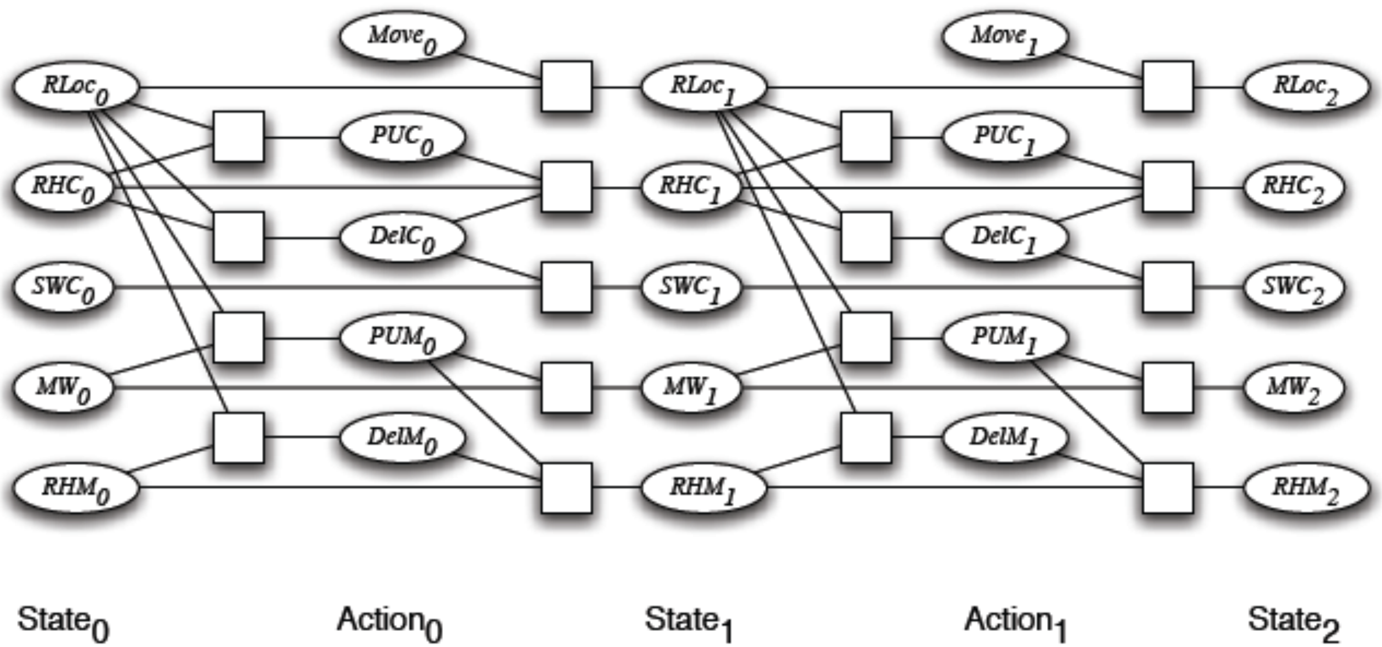
- We need to 'unroll the plan' for a fixed number of steps: this is called the horizon k
- To do this with a horizon of k:
  - construct a CSP variable for each STRIPS state variable at each time step from 0 to k
  - construct a boolean CSP variable for each STRIPS action at each time step from 0 to k - 1.



Initial State(s) and Goal(s)

– How can we represent the initial state(s) and the goal(s) with this representation?

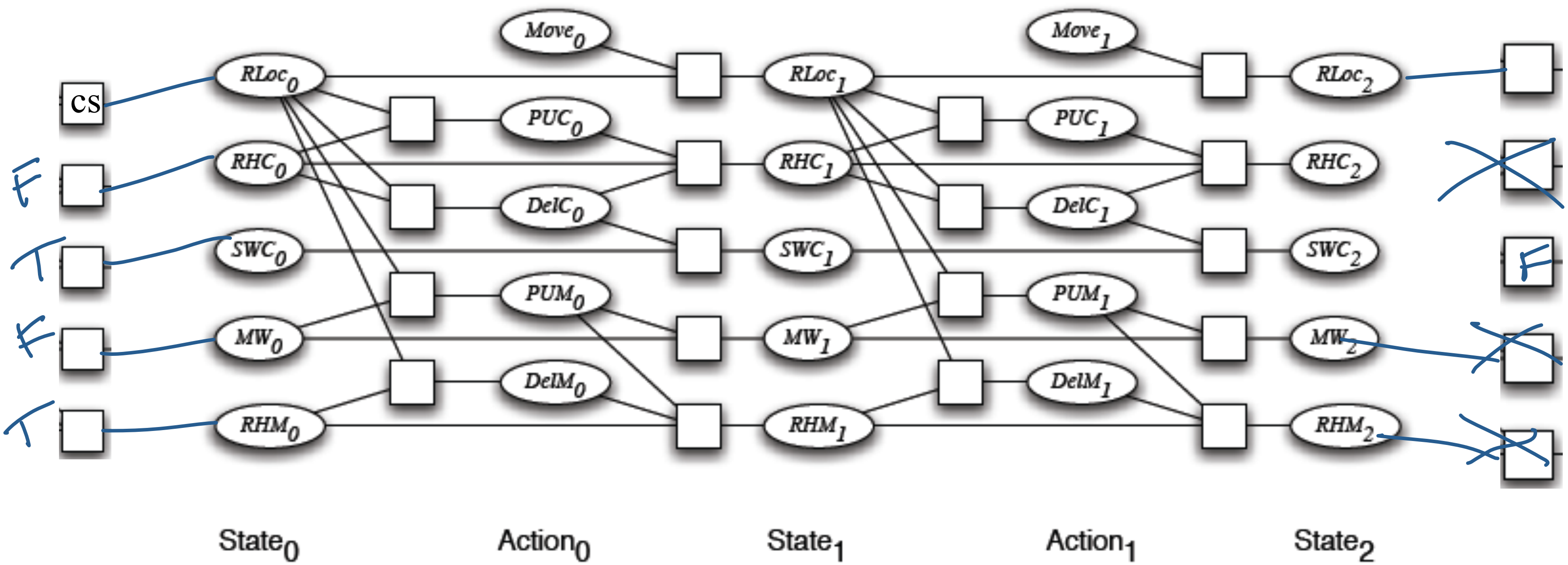
- e.g. Initial state with *Sam wanting coffee* and *Rob at the coffee shop, with no coffee and no mail*
- Goal: *Sam does not want coffee*



Initial and Goal Constraints

– initial state constraints: **unary** constraints on the values of the state variables at time 0

– goal constraints: **unary** constraints on the values of the state variables at time *k*

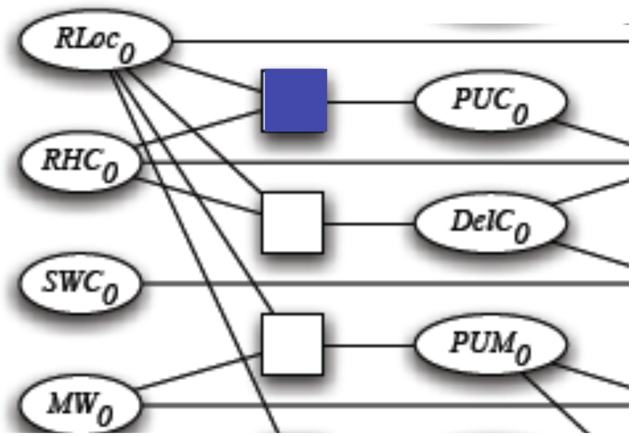


CSP Planning: Prec. Constraints

- As usual, we have to express the preconditions and effects of actions:
  - precondition constraints
    - hold between state variables at time t and action variables at time t
    - specify when actions may be taken

Rob Location

Rob has coffee



PUC_0

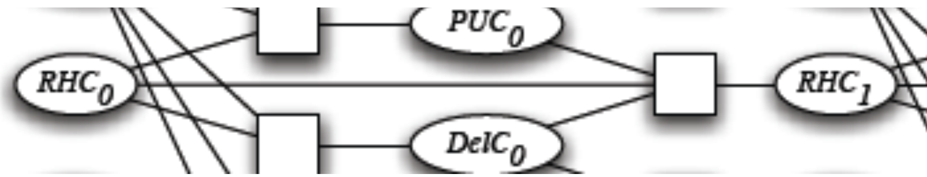
$RLoc_0$	RHC_0	PUC_0
cs	T	F
cs	F	T
cs	F	F
mr	*	F
lab	*	F
off	*	F

Need to allow for the option of *not* taking an action even when it is valid

CSP Planning: Effect Constraints

- Given a state at time t, and at time t+1, we want a constraint that involves all the actions that could potentially affect this state
  - For instance, let's consider RHC at time t and t+1

RHC_t	$DelC_i$	PUC_i	RHC_{t+1}
T	T	T	T
T	T	F	F
T	F	T	T
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F



CSP Planning: Solving the problem

Map STRIPS Representation for horizon 1, 2, 3, …, until solution found

Run arc consistency and search or stochastic local search!

k = 0

Is $State_0$ a goal?

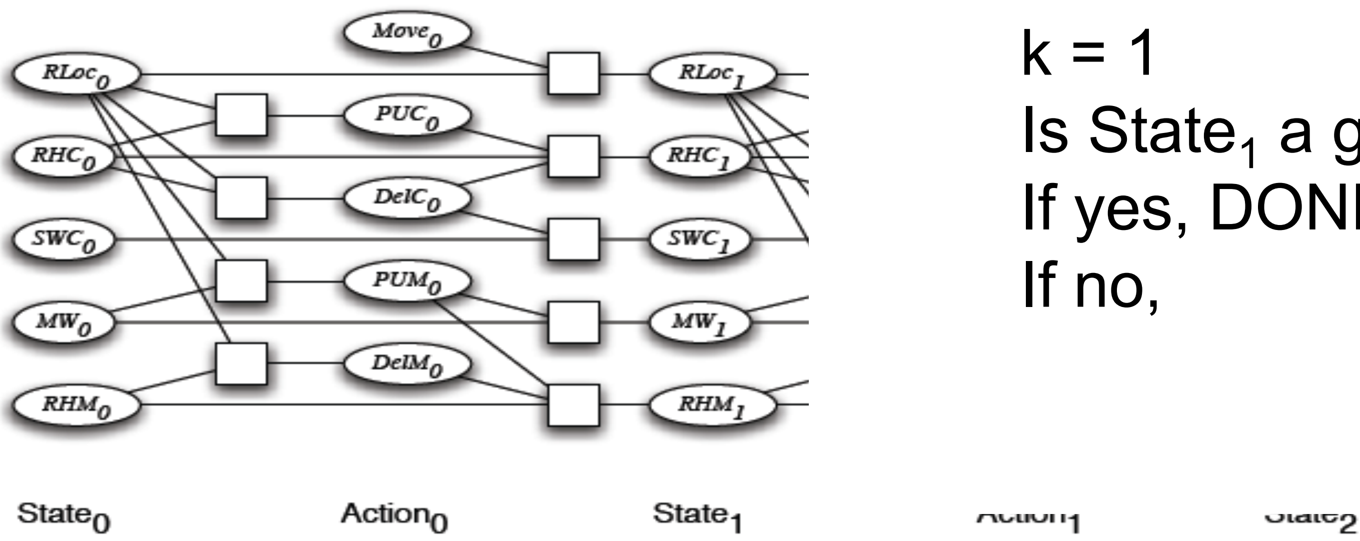
If yes, DONE!

If no,

CSP Planning: Solving the problem

Map STRIPS Representation for horizon k =1

Run arc consistency and search or stochastic local search!



k = 1

Is $State_1$ a goal

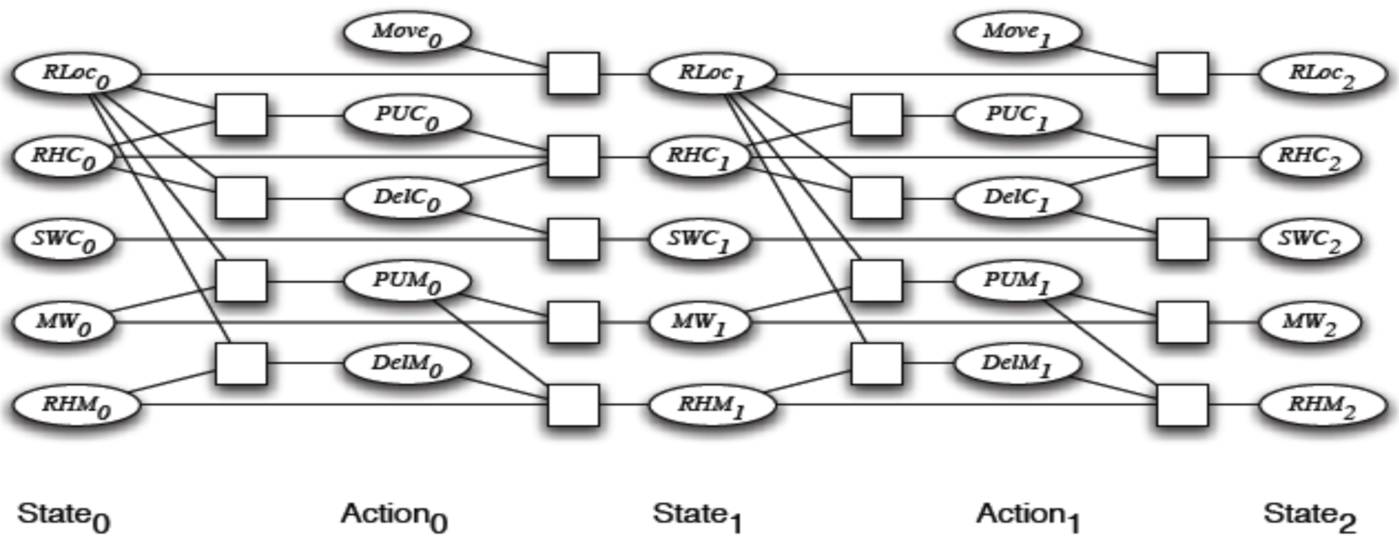
If yes, DONE!

If no,

CSP Planning: Solving the problem

Map STRIPS Representation for horizon k = 2

Run arc consistency, search, stochastic local search!



k = 2: Is $State_2$ a goal

If yes, DONE!

If no….continue

Solve Planning as CSP: pseudo code

```
solved = false
horizon = 0
While solved = false
    map STRIPS into CSP with horizon
    solve CSP -> solution
        if solution then
            solved = T
        else
            horizon = horizon + 1

Return solution
```

STRIPS to CSP applet

Allows you:

- to specify a planning problem in STRIPS
- to map it into a CSP for a given horizon
- the CSP translation is automatically loaded into the CSP applet where it can be solved

Under ‘Prototype Tools’ on the AIspace Home Page

Learning Goals for Planning

- Included in midterm
  - Represent a planning problem with the STRIPS representation
  - Explain the STRIPS assumption

- Excluded from midterm
  - Solve a planning problem by search (forward planning). Specify states, successor function, goal test and solution.
  - Construct and justify a heuristic function for forward planning
  - Translate a planning problem represented in STRIPS into a corresponding CSP problem (and vice versa)
  - Solve a planning problem with CSP by expanding the horizon