Planning: Representation and Forward Search

Computer Science cpsc322, Lecture 17

(Textbook Chpt 8.1 (Skip 8.1.1-2)- 8.2)

June, 1, 2017

Lecture Overview

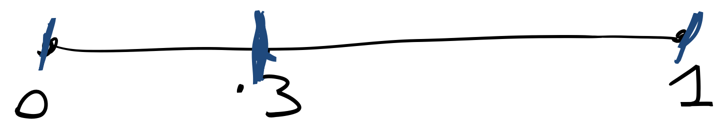
- **Clarification**
- **Where are we?**
- Planning
  - Example
  - STRIPS: a Feature-Based Representation
  - Forward Planning

Sampling a discrete probability distribution

e.g. Sim. Annealing. Select n' with probability P

P = .3

generate random number in [0,1]



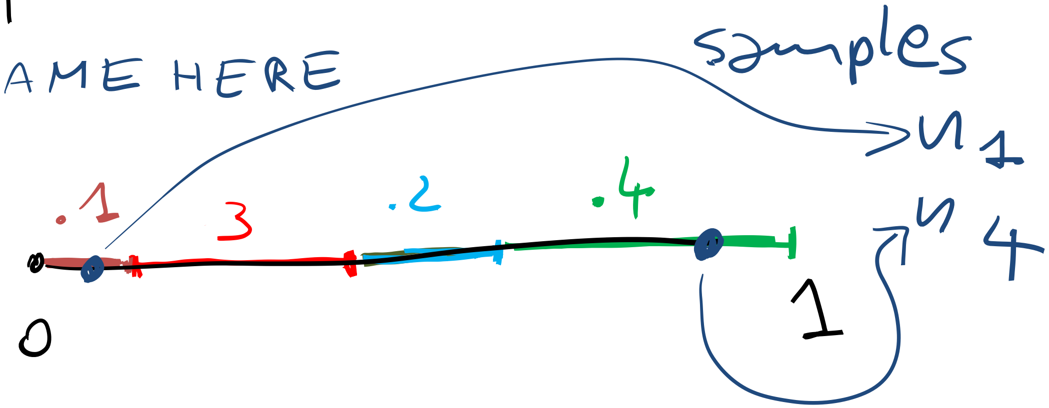
If < .3 accept n'

2

e.g. Beam Search: Select K individuals. Probability of selection proportional to their value

SAME HERE

n_1	$p_1 = .1$
n_2	$p_2 = .3$
n_3	$p_3 = .2$
n_4	$p_4 = .4$



Lecture Overview

- Clarifications
- **Where are we?**
- Planning
  - Example
  - STRIPS: a Feature-Based Representation
  - Forward Planning

Modules we'll cover in this course: R&Rsys

Environment

Problem		Deterministic	Stochastic
Static	Constraint Satisfaction	*Vars + Constraints* Arc Consistency Search → for CSP SLS	
Static	Inference	*Logics* Search → for Inference	*Belief Nets* Var. Elimination
Sequential	Planning	*STRIPS* Search → for complex planning	*Decision Nets* Var. Elimination *Markov Processes* Value Iteration

Representation

Reasoning Technique

Standard Search vs. Specific R&R systems

Constraint Satisfaction (Problems)(**A**): (**B**: domain splitting ……..)

- 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
- Successor function
- Goal test
- Solution
- Heuristic function (next class)

Inference

- State
- Successor function
- Goal test
- Solution
- Heuristic function

Lecture Overview

- Clarifications
- Where are we?
- Planning
  - Example
  - STRIPS representation and assumption
  - Forward Planning

Planning as Search: State and Goal

How to select and organize a sequence of actions to achieve a given goal…

State: Agent is in a possible world (**full assignments** to a set of variables/features)

A B C

domain (true, false) (T, F) (1 0)

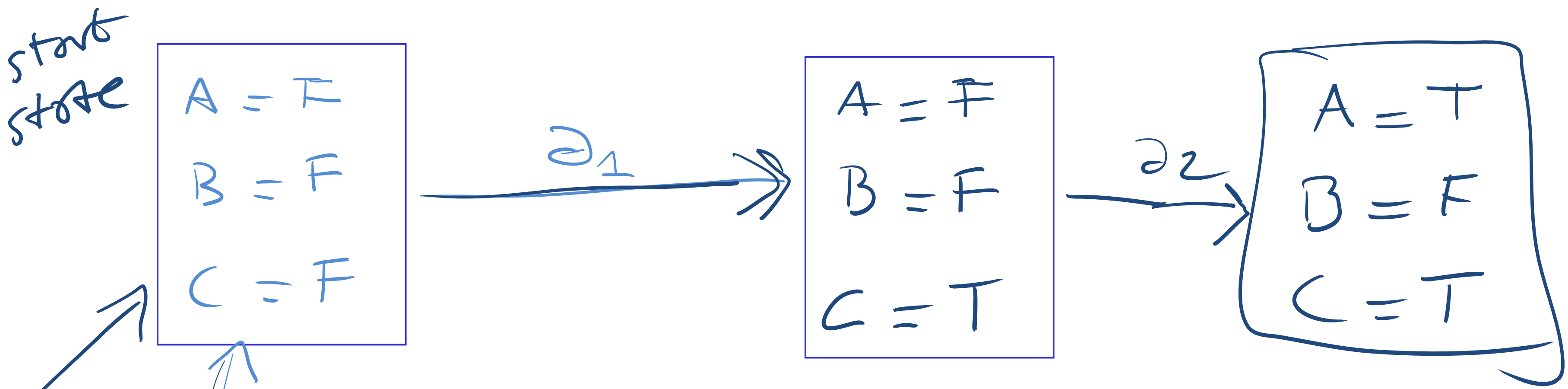
[A = T B = F C = T] sample state

Goal: Agent wants to be in a possible world were **some** variables are given specific values

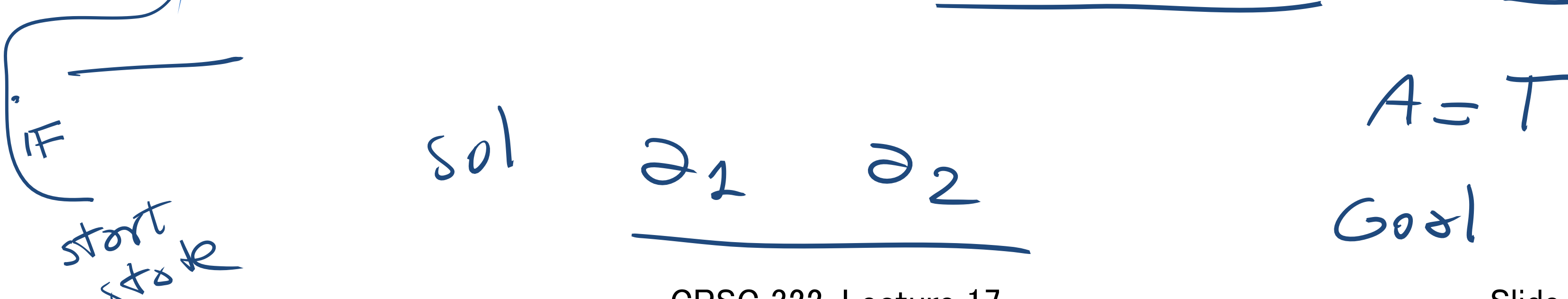
sample goal [A = T C = F]

Planning as Search: Successor function and Solution

Actions : take the agent from one state to another



Solution: sequence of actions that when performed will take the agent from the current state to a goal state

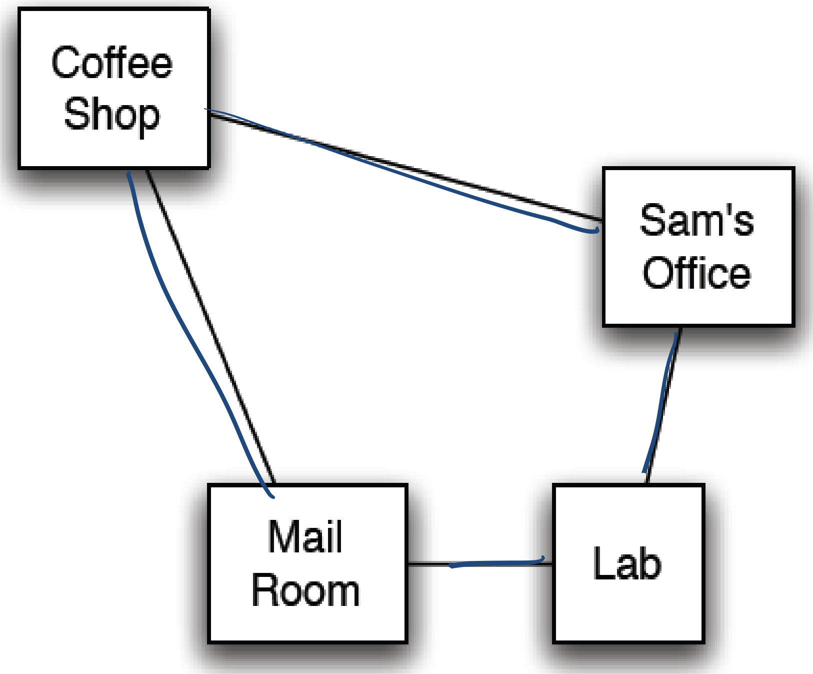


Lecture Overview

- Clarifications
- Where are we?
- Planning
  - **Example**
  - STRIPS representation and assumption
  - Forward Planning

Delivery Robot Example (textbook)

Consider a **delivery robot named Rob**, who must navigate the following environment, can deliver coffee and mail to Sam



Another example will be available as a Practice Exercise: "Commuting to UBC"

Delivery Robot Example: States

The state is defined by the following variables/features:

RLoc – Rob's location

- domain: coffee shop (*cs*), Sam's office (*off*), mail room (*mr*), or laboratory (*lab*)

RHC – Rob has coffee True/False.

SWC – Sam wants coffee T/F

MW – Mail is waiting T/F

RHM – Rob has mail T/F

RHC=T rhc

RHC=F $\overline{rhc}$

two different notations

4 2 2 2 2

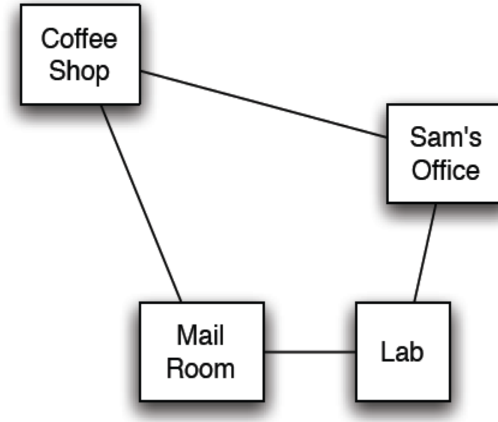
Example state: {cs, rhc, $\overline{swc}$, $\overline{mw}$, rhm}

Number of states: 64

iclicker.

A. 32 B. 64 C. 48 D. 16

Delivery Robot Example: Actions



The robot's *actions* are:

Move – Rob's move action

- move clockwise (*mc*), move anti-clockwise (*mac*) not ~~move (*mm*)~~

PUC – Rob picks up coffee

- must be at the coffee shop

preconditions

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

Lecture Overview

- Clarifications
- Were are we?
- Planning
  - Example
  - STRIPS representation and assumption (STanford Research Institute Problem Solver) 70's 80's
  - Forward Planning

STRIPS action representation

The key to sophisticated planning is modeling actions

In STRIPS, an action has two parts:

1. Preconditions: a set of assignments to features that **must be satisfied** in order for the action to be legal
2. Effects: a set of assignments to features that are **caused** by the action

STRIPS actions: Example S

STRIPS representation of the action pick up coffee, *PUC* :

- **preconditions** *Loc* = *cs* and *RHC* = F
- **effects** *RHC* = T

STRIPS representation of the action deliver coffee, *DelC* :

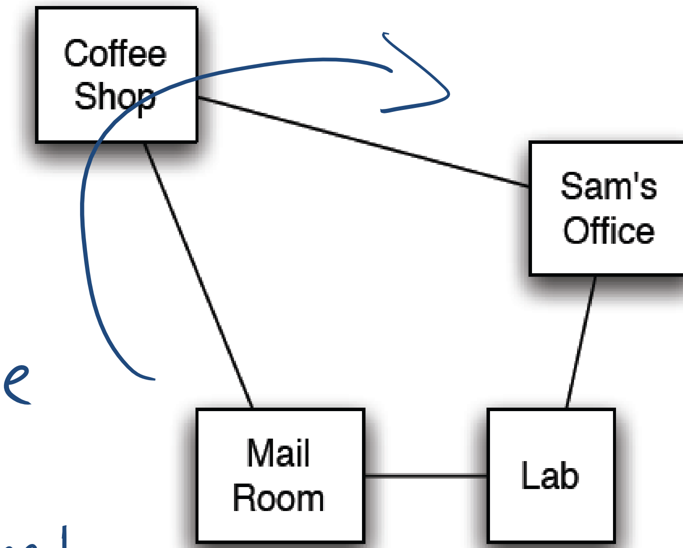
- **preconditions** *Loc* = off and *RHC* = T (SWC = T)
- **effects** RHC = F and SWC = F

Note in this domain Sam doesn't have to want coffee for Rob to deliver it; one way or another Sam doesn't want coffee after delivery

STRIPS actions: MC and MAC

STRIPS representation of the action

MoveClockwise ?



mc-cs
- Prec loc=cs
- Eff loc=office

mc-off

⋮

→ you need 4 specific actions

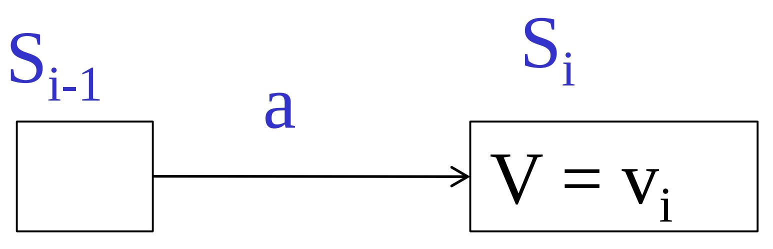
mac-cs

→ and 4 specific actions for move anticlockwise

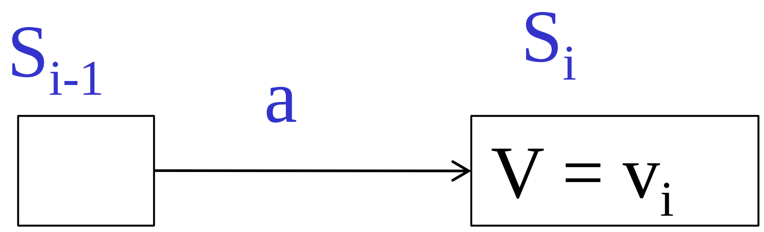
STRIPS Actions (cont')

The STRIPS assumption: all features not explicitly changed by an action stay unchanged

- So if the feature **V** has value v_i in state S_i , after action **a** has been performed,
  - what can we conclude about **a** and/or the state of the world S_{i-1} ,immediately preceding the execution of **a**?



What can we conclude about *a* and/or the state of the world S_{i-1} ,immediately preceding the execution of *a*?



A. $V = v_i$ was TRUE in S_{i-1}

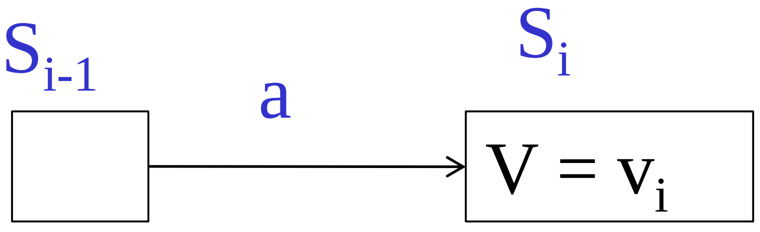
B. One of the effects of **a** is to set $V = v_i$

C. At least one of the above

D None of the above

what can we conclude about *a* and/or the state of the world S_{i-1} ,immediately preceding the execution of *a*?

3 At least one of the above



Lecture Overview

- Clarifications
- Where are we?
- Planning
  - Example
  - STRIPS representation and assumption (STanford Research Institute Problem Solver)
  - Forward Planning

Forward Planning

To find a plan, a solution: search in the state-space graph.

- The **states** are the **possible worlds**
- The **arcs** from a state ***s*** represent all of the **actions** that are legal in state ***s***.
- A **plan** is a path from the state representing the initial state to a state that satisfies the goal.

What actions **a** are legal/possible in a state **s**?

A. Those where **a**'s effects are satisfied in **s**

B. Those where **a**'s preconditions are satisfied in **s**

C. Those where the state **s'** reached via **a** is on the way to the goal

Forward Planning

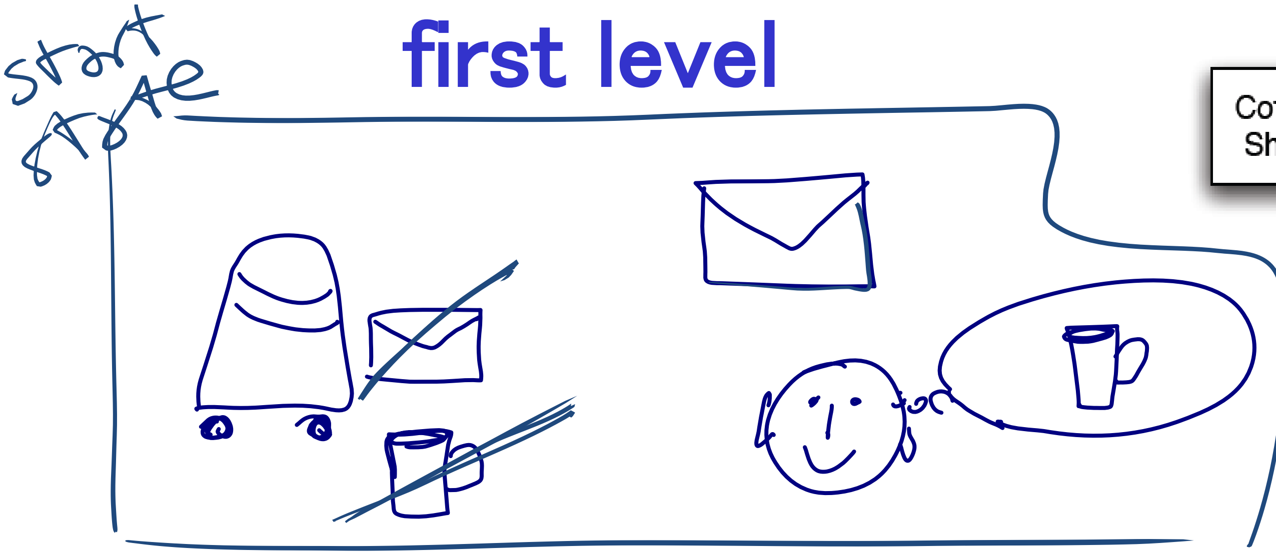
To find a plan, a solution: search in the state-space graph.

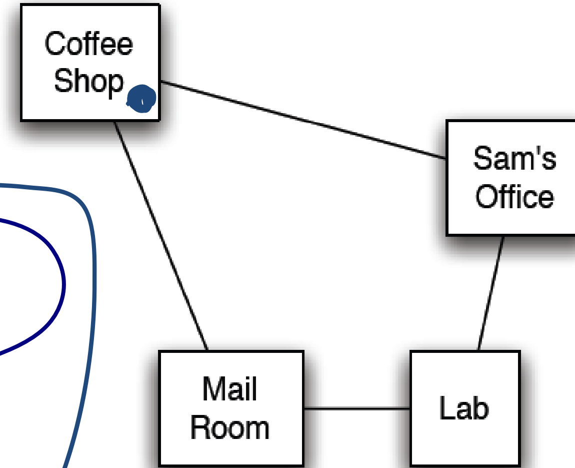
- The **states** are the **possible worlds**
- The **arcs** from a state ***s*** represent all of the **actions** that are legal in state ***s***.
- A **plan** is a path from the state representing the initial state to a state that satisfies the goal.

What actions *a* are legal/possible in a state s?

Those where **a**'s preconditions are satisfied in **s**

Example state-space graph: first level





Actions

mc: move clockwise

mac: move anticlockwise

} Just two to simplify the example

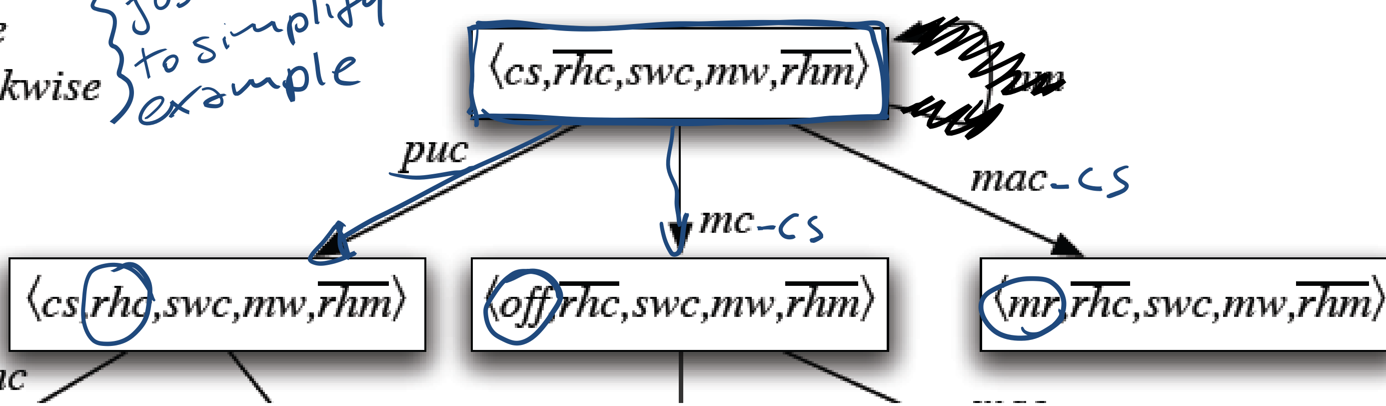
[redacted]

puc: pick up coffee

dc: deliver coffee

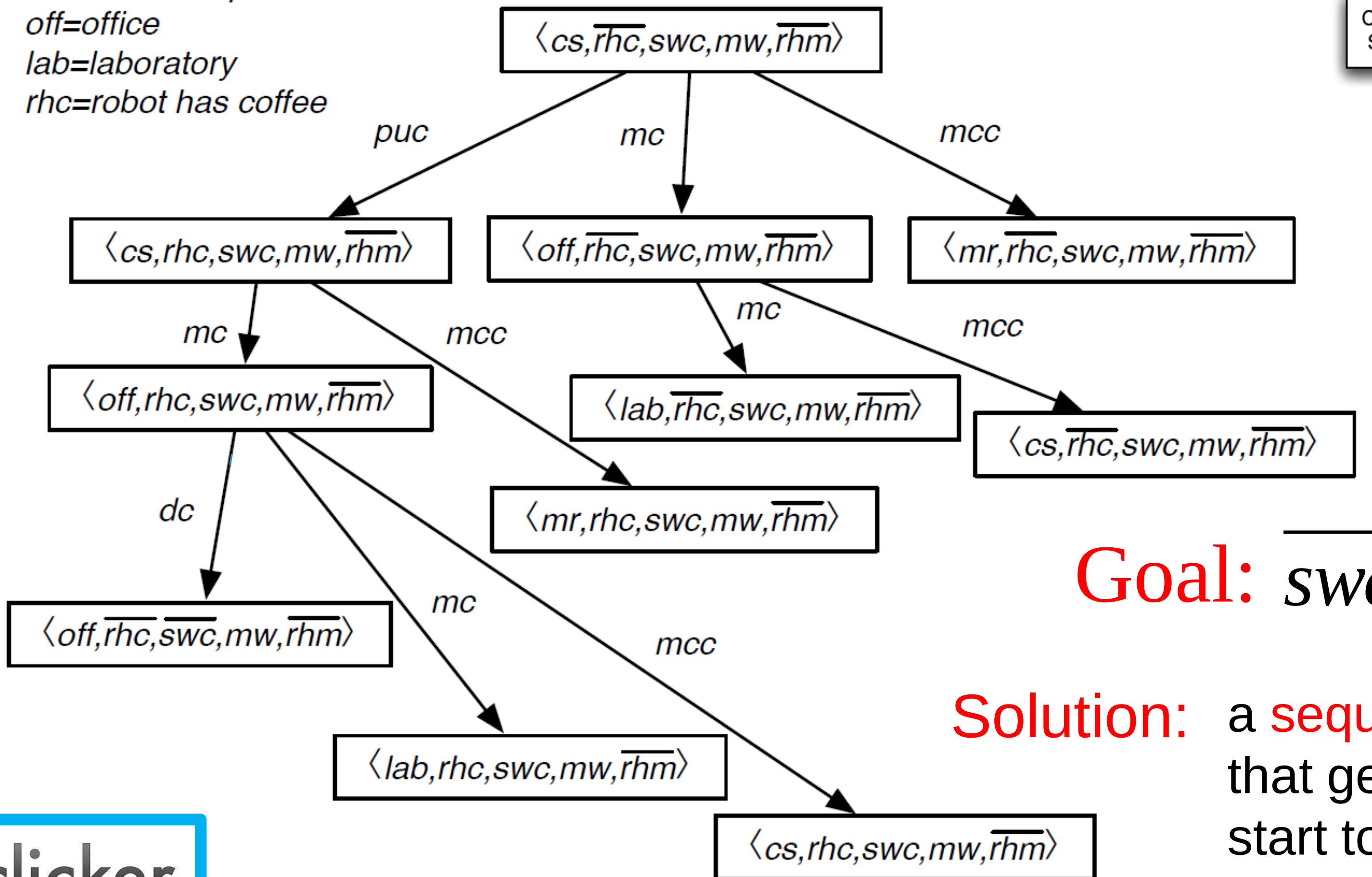
pum: pick up mail

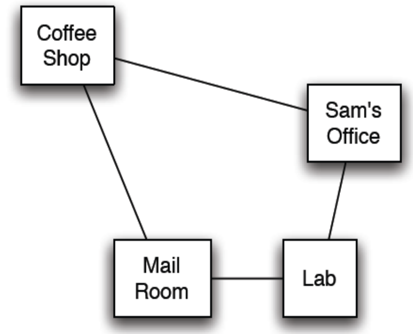
dm: deliver mail



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

iclicker.

What is a solution to this planning problem?

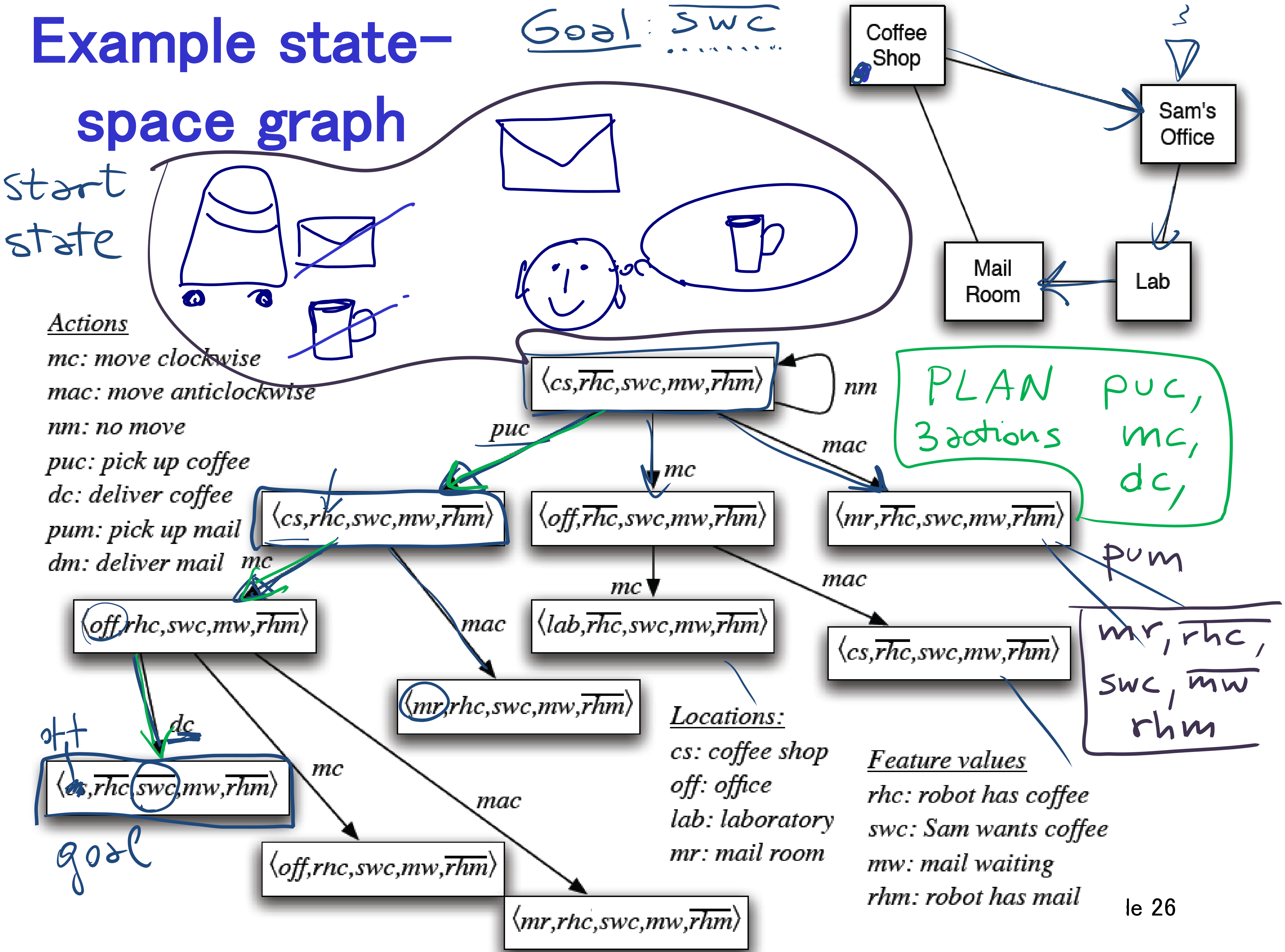
A. (puc, mc)

B. (puc, mc, mc)

C. (puc, dc)

D. (puc, mc, dc)

Example state-space graph



Learning Goals for today' s class

You can:

- Represent a planning problem with the **STRIPS representation**
- Explain the **STRIPS assumption**
- Solve a planning problem by search (**forward planning**). Specify states, successor function, goal test and solution.

Next class

Finish Planning (Chp 8)

- Heuristics for planning (*not on textbook*)
- Mapping planning problem into a CSP (8.4)

Course Announcements

- Start working on Assignment2 (CSP) – due June 8
- Work on Practice Exercises (under Aispace)