Search: Intro

Computer Science cpsc322, Lecture 4

(Textbook Chpt 3.0-3.4)

Sept, 12, 2012

Office Hours

- **Giuseppe Carenini** (carenini@cs.ubc.ca; office CICSR 105)

Teaching Assistants

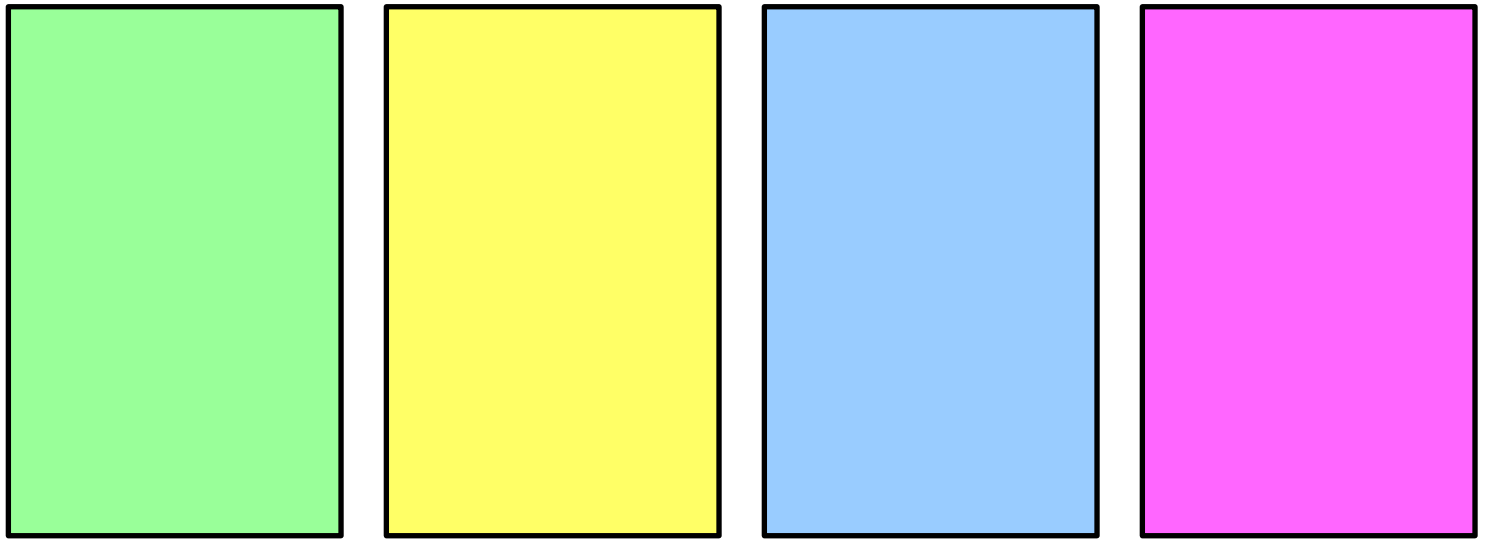
- **Nathan Tomer** ntomer@cs.ubc.ca

- **Tatsuro Oya** toya@cs.ubc.ca

- **Seyed M. Kazemi** smkazemi@cs.ubc.ca

Colored Cards

- You need to have 4 colored index cards
  - Come and get them from me if you still don't have them



- You will use these as voting cards
  - Cheap low tech variant of clickers

Please bring them to class every time

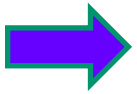
“Deterministic agent” means an agent that

- Has perfect knowledge of its environment
- Has perfect knowledge of the effect that its actions can have on the environment
- Both of the above
- None of the above

"Deterministic agent" means an agent that

Has perfect knowledge of its environment

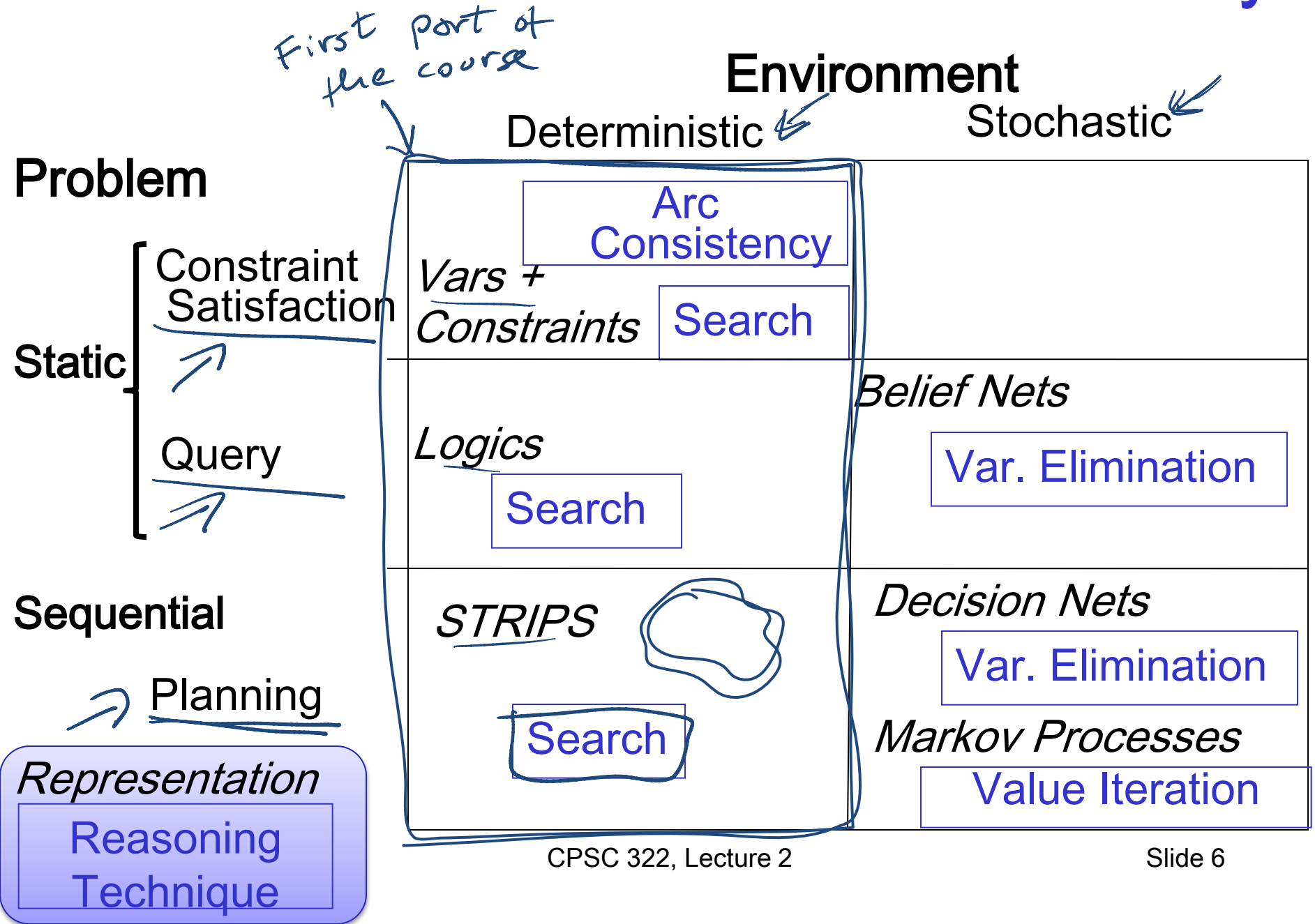
Has perfect knowledge of the effect that its actions can have on the environment



Both of the above

None of the above

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



Lecture Overview

- **Simple Agent and Examples**
- Search Space Graph
- Search Procedure

Simple Planning Agent

Deterministic, goal-driven agent

- Agent is in a **start state**
- Agent is given a **goal** (subset of possible states)
- Environment changes only when the agent acts
- Agent perfectly knows:
  - **what actions can be applied** in any given state
  - **the state it is going to end up** in when an action is applied in a given state
- The sequence of actions and their appropriate ordering is the **solution**

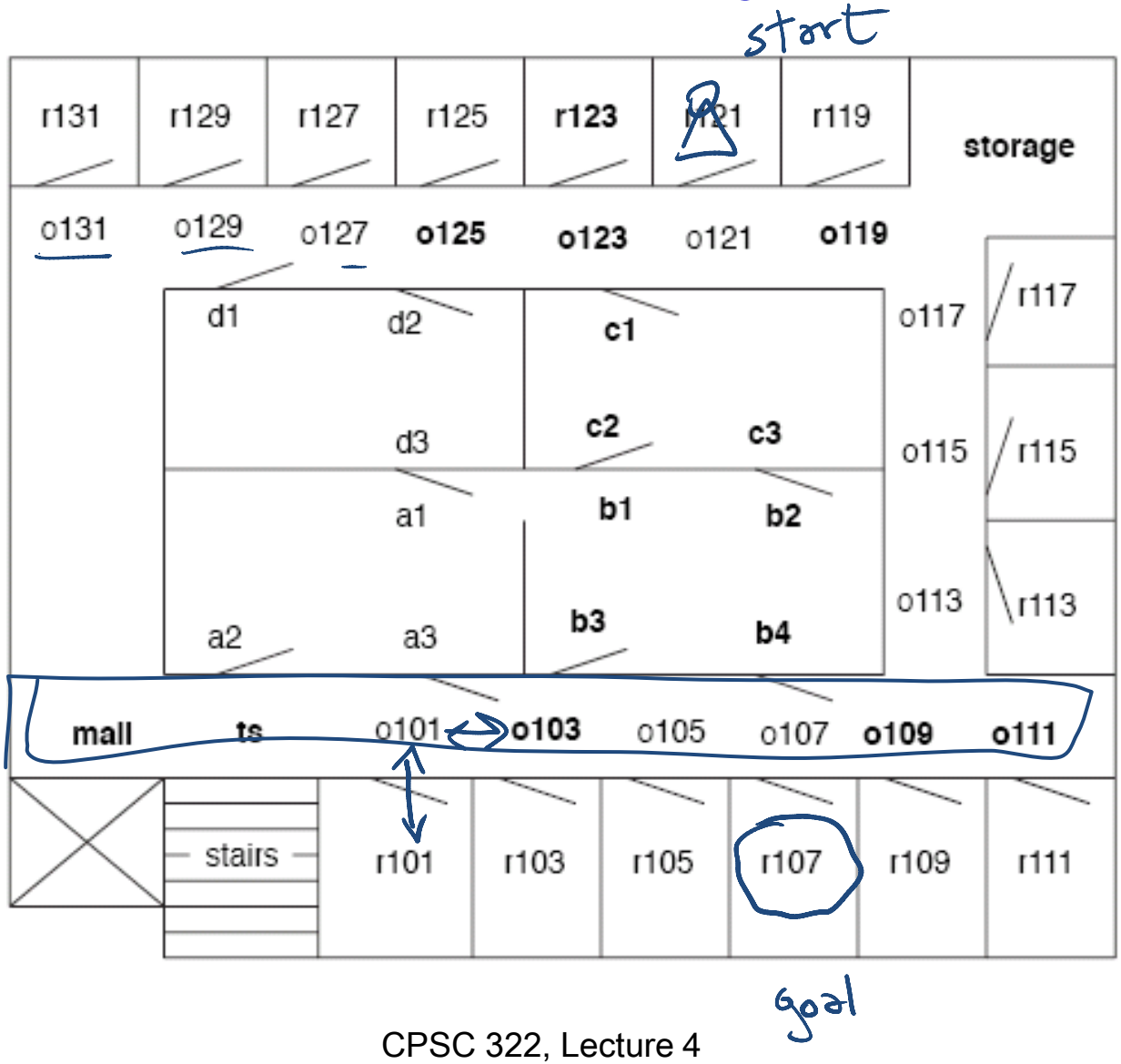
Three examples

1. A delivery robot planning the route it will take in a bldg. to get from one room to another

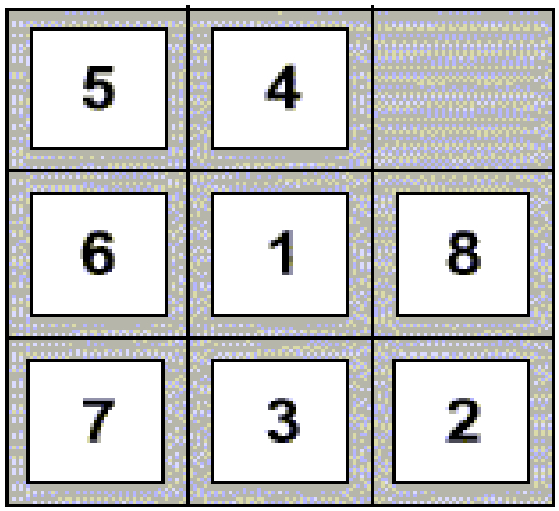
2. Solving an 8-puzzle

3. Vacuum cleaner world

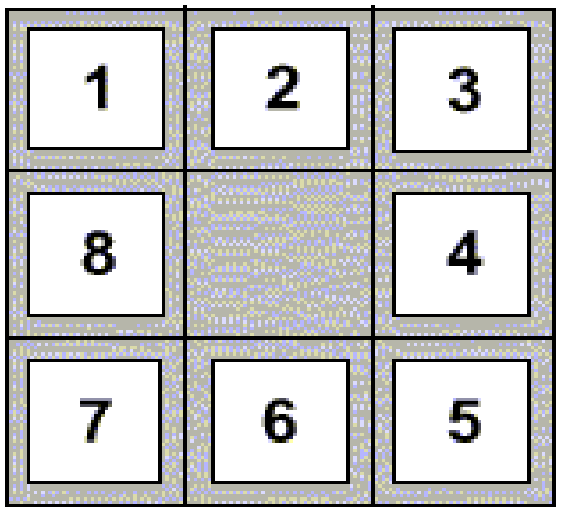
Example1: Delivery Robot



Eight Puzzle



Start State



Goal State

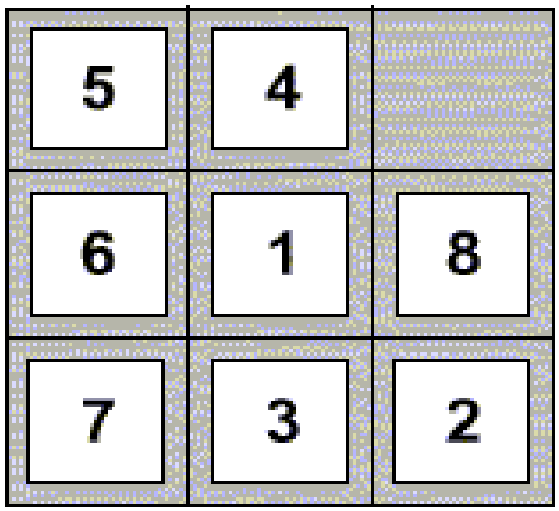
States: each state specifies which number/blank occupies each of the 9 tiles

HOW MANY STATES ? 8^9 2^9 9^9 $9!$

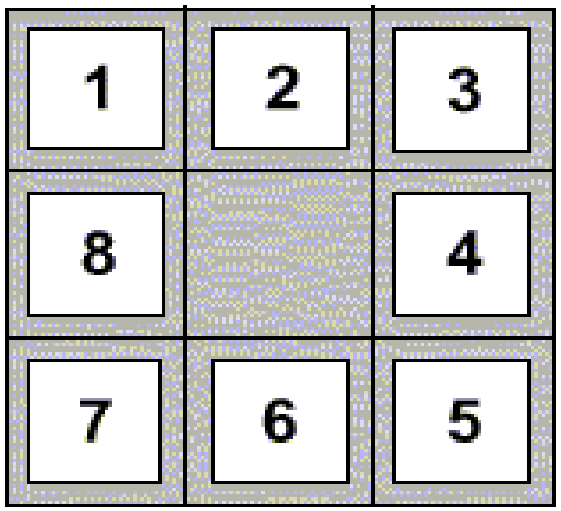
Operators:

Goal:

Eight Puzzle



Start State



Goal State

States: each state specifies which number/blank occupies each of the 9 tiles

HOW MANY STATES ? *9!*

Operators: blank moves left, right, up down

Goal: configuration with numbers in right sequence

Example 2: 8-Puzzle?

of states

9!

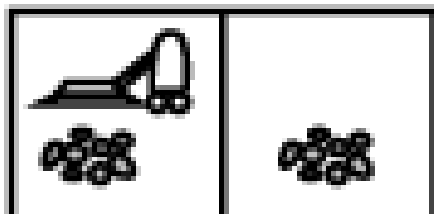
$\sim 360 \times 10^3$

Possible start state

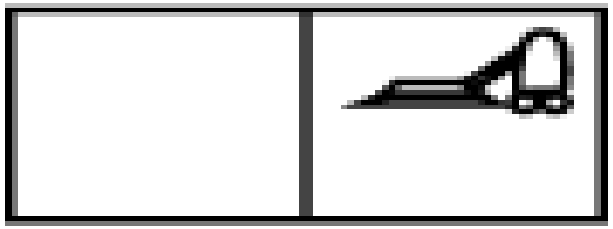
Goal state

Example: vacuum world

- States
  - Two rooms: r1, r2
  - Each room can be either dirty or not
  - Vacuuming agent can be in either in r1 or r2



Possible start state



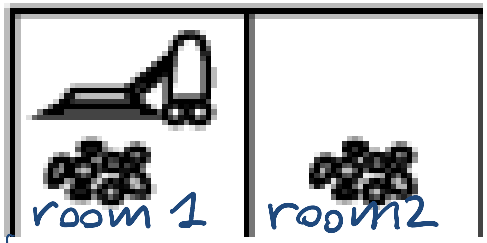
Possible goal state

Example: vacuum world

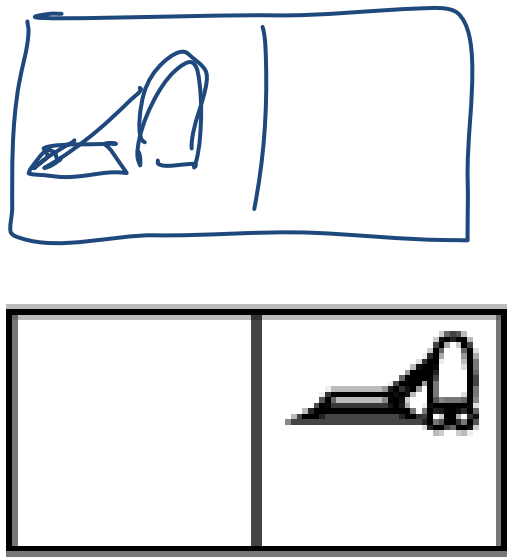
loc 2 values $\{r_1, r_2\}$

r_1-clean T/F

r_2-clean



Possible start state



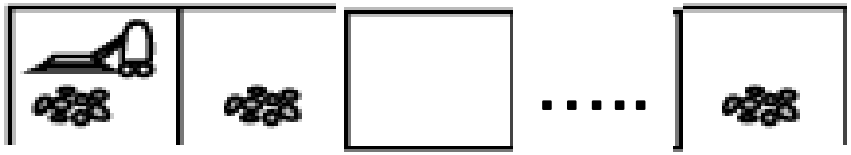
Goal state

# of states

2^2

2^2

can be subset of states



Suppose we have the same problem with k rooms.

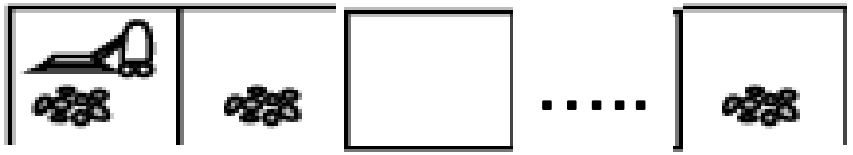
The number of states is….

k^3

$k * 2k$

$k * 2^k$

$2 * k^k$



Suppose we have the same problem with k rooms.

The number of states is….

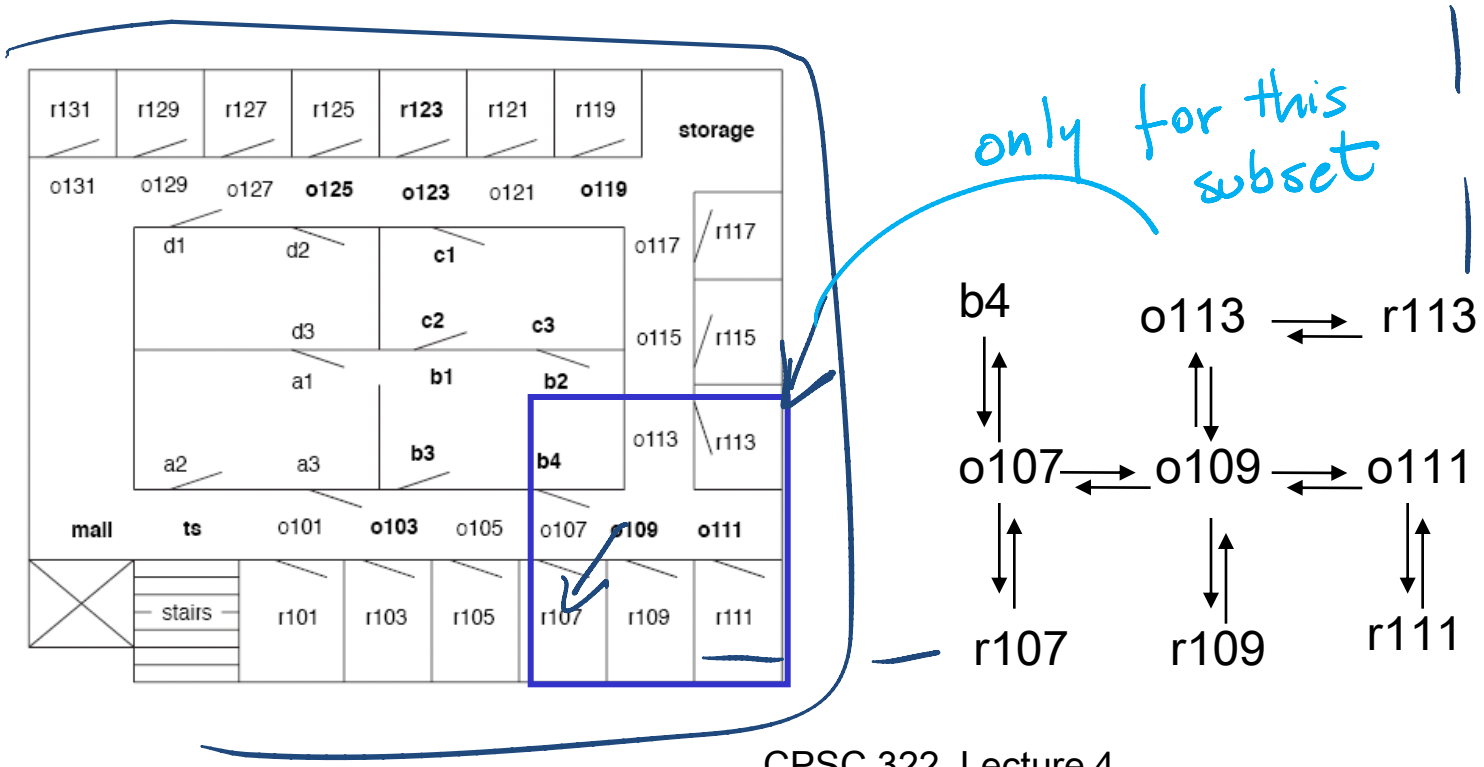
$k * 2^k$

Lecture Overview

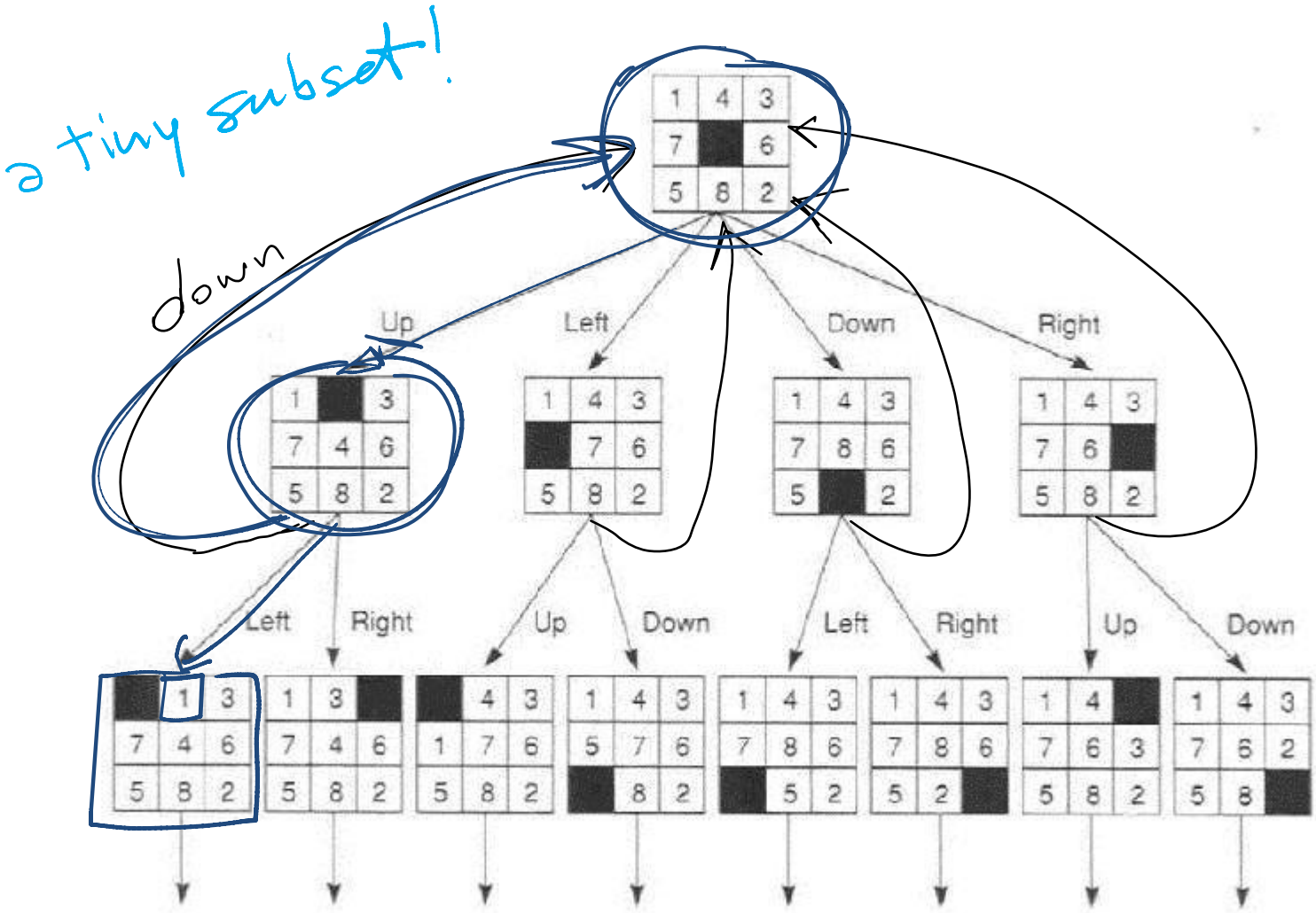
- Simple Agent and Examples
- Search Space Graph
- Search

How can we find a solution?

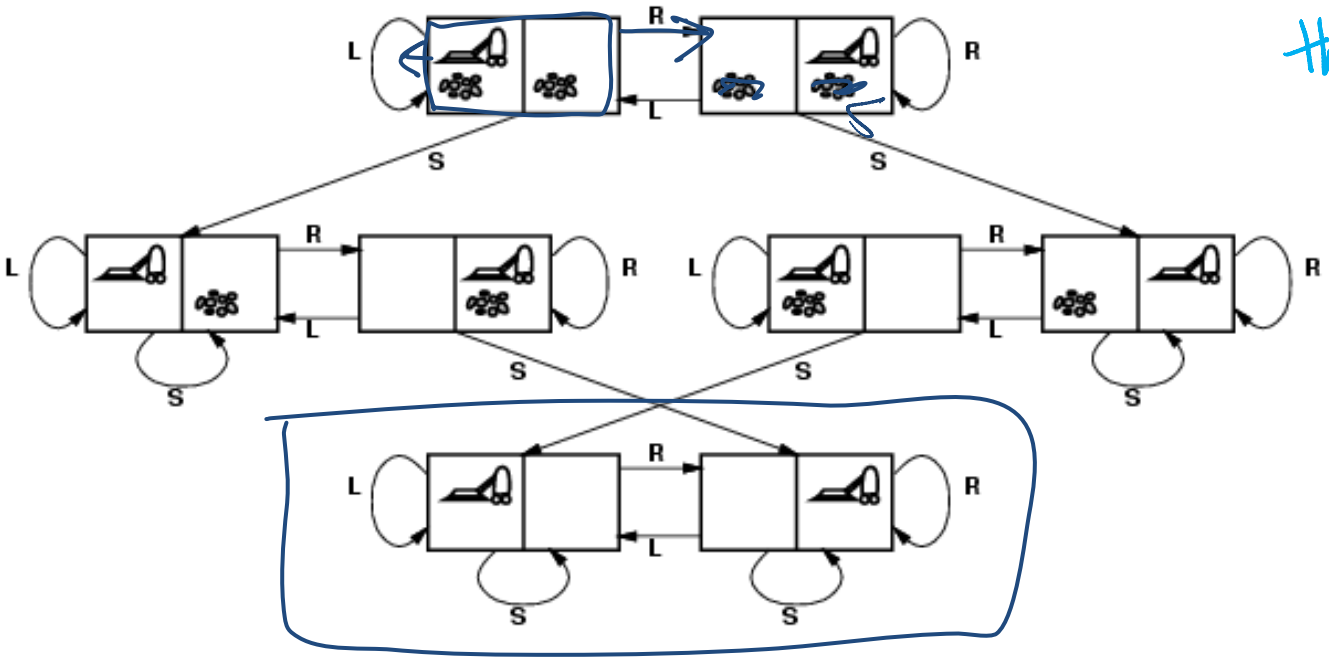
- How can we find a sequence of actions and their appropriate ordering that lead to the goal?
- Define underlying **search space graph** where **nodes are states** and **edges are actions**.



Search space for 8puzzle



Vacuum world: Search space graph



the whole space

states? Where it is dirty and robot location

actions? *Left*, *Right*, *Suck*

Possible goal test? no dirt at all locations

Lecture Overview

- Simple Agent and Examples
- State Space Graph
- **Search Procedure**

Search: Abstract Definition

How to search

- Start at the start state
- Consider the effect of taking different actions starting from states that have been encountered in the search so far
- Stop when a goal state is encountered

To make this more formal, we'll need review the **formal definition of a graph**...

Search Graph

A ***graph*** consists of a set N of ***nodes*** and a set A of ordered pairs of nodes, called ***arcs***.

Node n_2 is a ***neighbor*** of n_1 if there is an arc from n_1 to n_2. That is, if $\langle n_1, n_2 \rangle \in A$.

A ***path*** is a sequence of nodes $n_0, n_1, n_2, \ldots, n_k$ such that $\langle n_{i-1}, n_i \rangle \in A$.

A ***cycle*** is a non-empty path such that the start node is the same as the end node

A ***directed acyclic graph*** (DAG) is a graph with no cycles

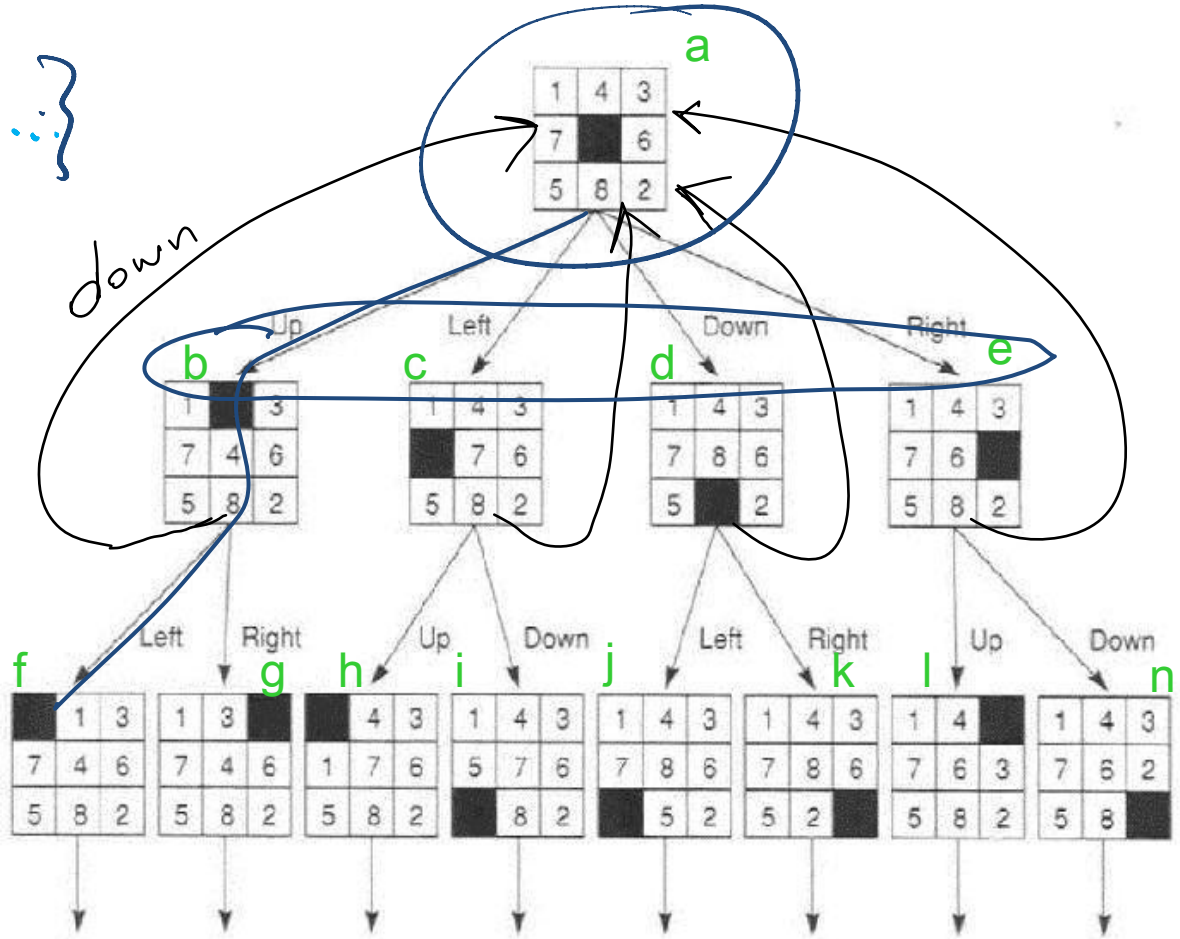
Given a start node and goal nodes, a ***solution*** is a path from a start node to a goal node.

Examples for graph formal def.

$N = \{ a\ b\ \ldots \}$

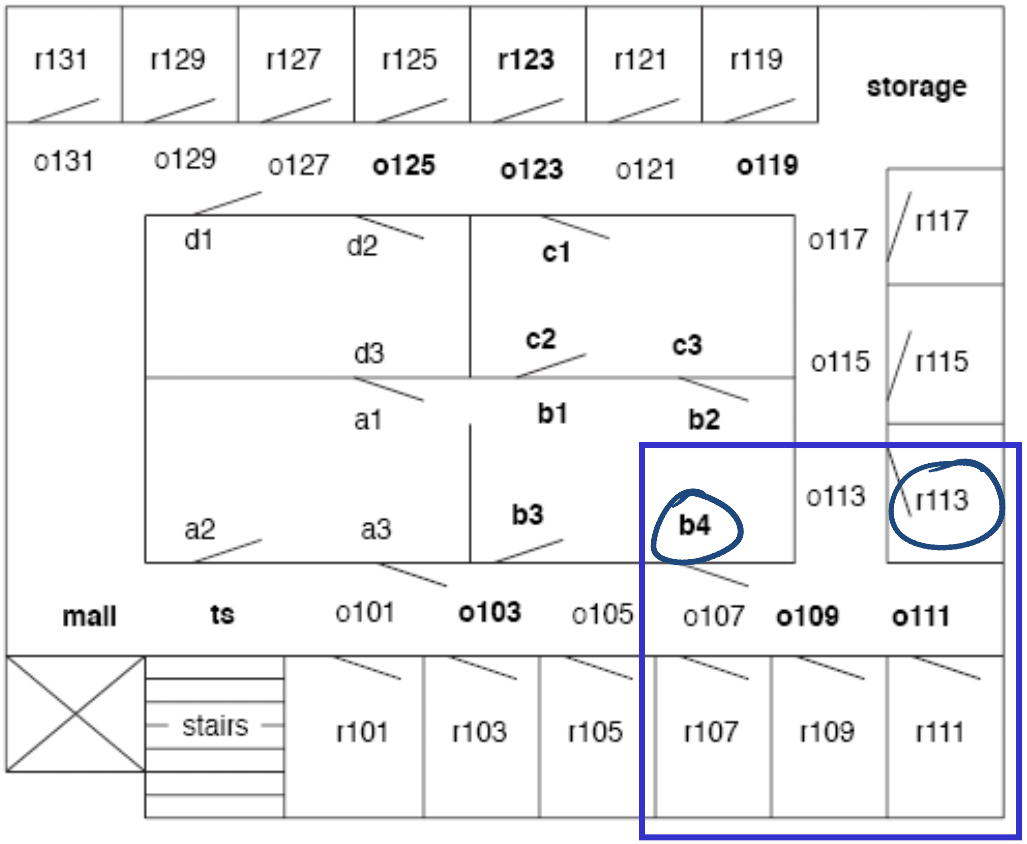
$A = \{ \langle a,b \rangle \langle a,c \rangle \ldots \}$

. . . .

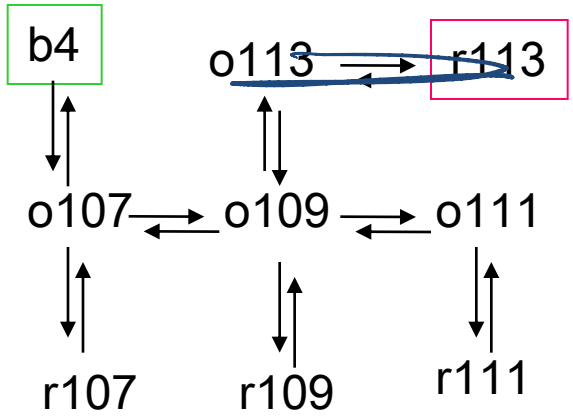


Examples of solution

- Start state b4, goal r113
- Solution <b4, o107, o109, o113, r113>



but there are many others!



Graph Searching

Generic search algorithm: given a graph, start node, and goal node(s), incrementally explore paths from the start node(s).

Maintain a **frontier of paths** from the start node that have been explored.

As search proceeds, the frontier expands into the unexplored nodes until (hopefully!) a goal node is encountered.

The way in which the frontier is expanded defines the search strategy.

Generic Search Algorithm

Input: a graph, a start node, Boolean procedure *goal(n)* that tests if *n* is a goal node

frontier:= [<*s*>: *s* is a start node];

While *frontier* is not empty:

  select and **remove** path <n_o,….,n_k> from *frontier;*

  If *goal*(n_k)

    return <n_o,….,n_k>;

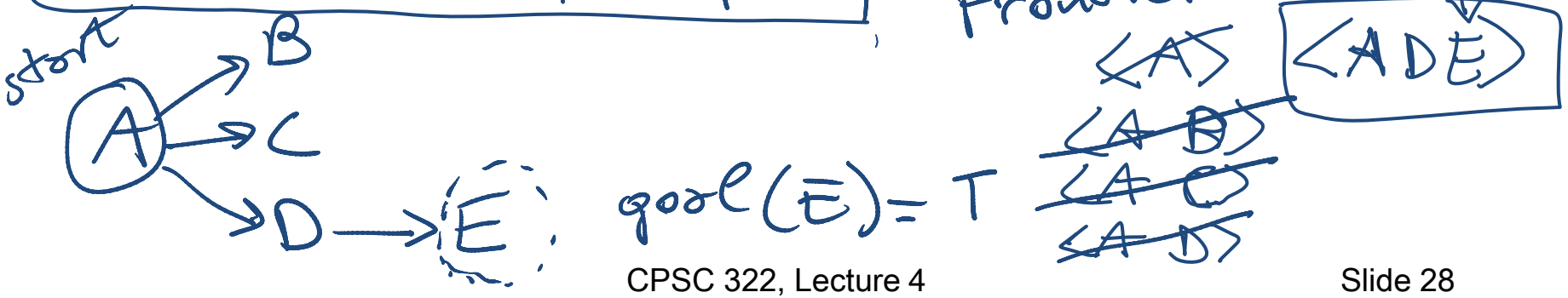
  For every neighbor *n* of n_k

    add <n_o,….,n_k, *n*> to *frontier;*

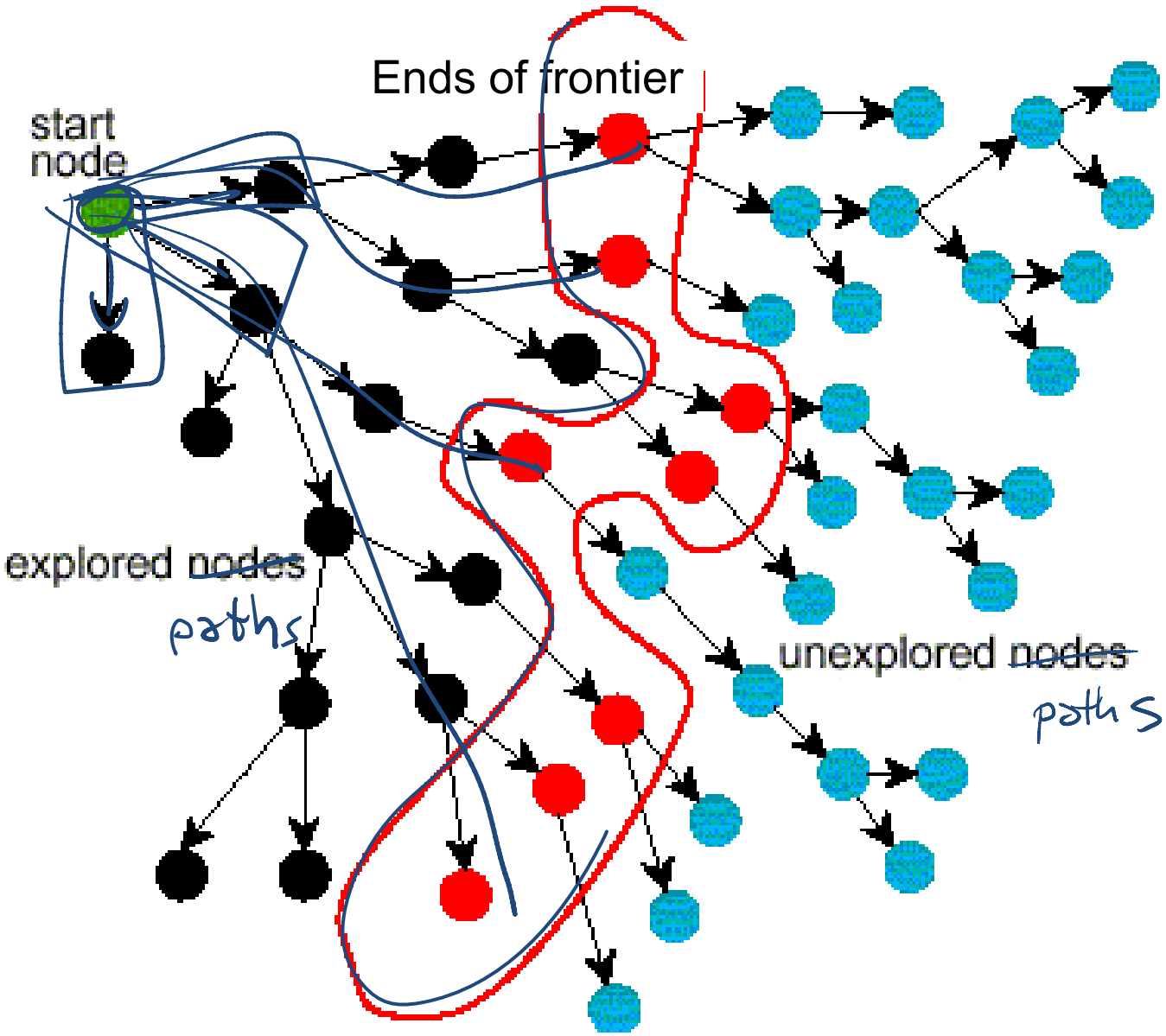
end

with cycles may get into infinite loop

no solution found



Problem Solving by Graph Searching

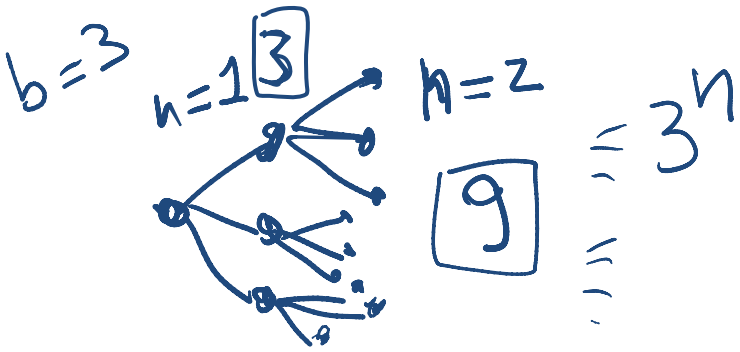


Branching Factor

The *forward branching factor* of a node is the number of arcs going out of the node

The *backward branching factor* of a node is the number of arcs going into the node

If the forward branching factor of any node is *b* and the graph is a tree, how many nodes are *n* steps away from a node?



nb b^n n^b n/b

Lecture Summary

- Search is a key computational mechanism in many AI agents
- We will study the basic principles of search on the simple **deterministic planning agent model**

Generic search approach:

- define a search space graph,
- start from current state,
- incrementally explore paths from current state until goal state is reached.

The way in which the frontier is expanded defines the search strategy.

Learning Goals for today's class

- **Identify** real world examples that make use of deterministic, goal-driven planning agents
- **Assess** the size of the search space of a given search problem.
- **Implement** the generic solution to a search problem.

How many possible states

see also Mars Explorer Lecture 2

Next class (Fri)

- **Uninformed search** strategies

(read textbook Sec. 3.5)

- First **Practice Exercise** 3.A
- http://www.aispace.org/exercises.shtml