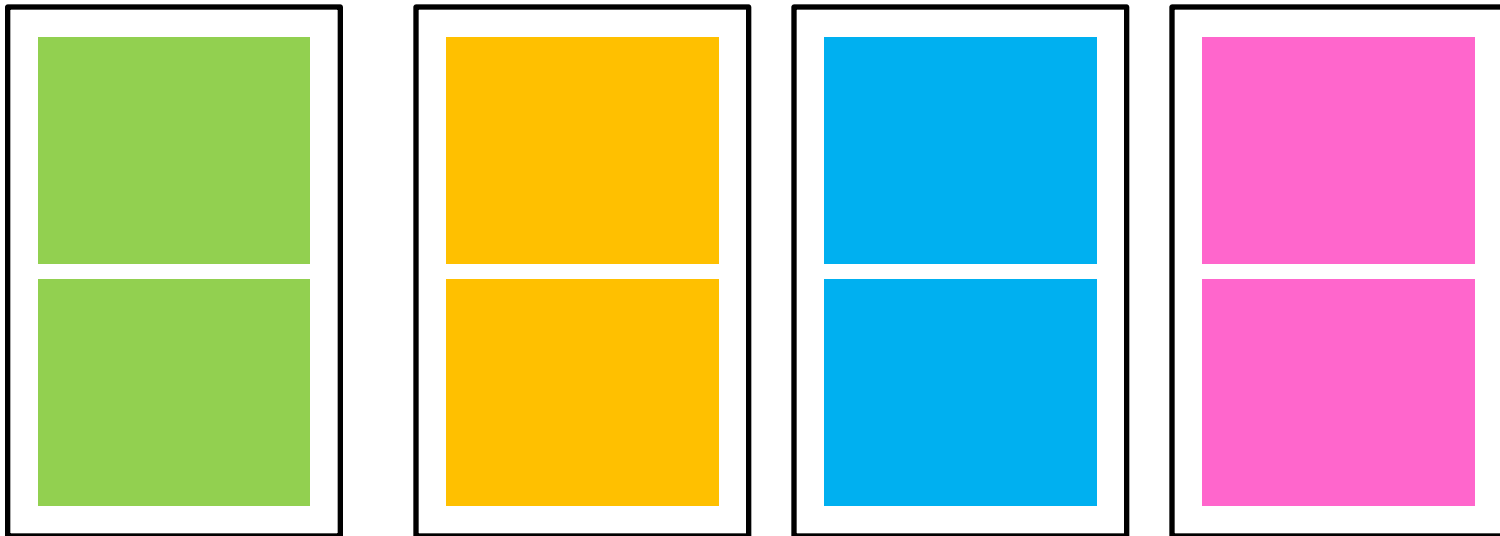# Search: Representation and General Search Procedure

CPSC 322 – Search 1

January 12, 2011
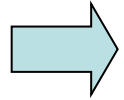
Textbook §3.0 – 3.4

# Colored Cards: Do you Have Them?

- If not, please come to the front and pick up
  - 4 index cards
  - 2 Post-it per colour (Blue, Yellow, Green, Pink)
- Use this material to make 4 "voting cards" as below
  - Low budget variant of clickers
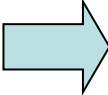- Please bring them to class every time

# Today's Lecture

→ Search: motivation

- Simple Search Agent and Examples

- General Search Procedure

# Course Map

| Dimensions / Course Modules | Deterministic vs. Stochastic | Static vs. Sequential | States vs. Features vs. Relations |
|---|---|---|---|
| 1. Search | Deterministic | Static | States |
| 2. CSPs | Deterministic | Static | Features |
| 3. Planning | Deterministic | Sequential | States or Features |
| 4. Logic | Deterministic | Static | Relations |
| 5. Uncertainty | Stochastic | Static | Features |
| 6. Decision Theory | Stochastic | Sequential | Features |

# Search is a powerful tool to know well

- World champion AI programs:
  - Checkers 1994 (unbeatable!)
  - Chess 1997


- Can also be used to solve
  - Constraint satisfaction problems (CSP), course module 2
  - Constraint optimization problems
    - Very important in operations research and in industry
  - Planning, course module 3
  - Problems in game AI
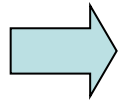    - Infinite Mario Bros: http://www.youtube.com/watch?v=0s3d1LfjWCI
  - …

# Search for Playing Infinite Mario Bros



http://www.youtube.com/watch?v=0s3d1LfjWCI

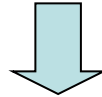# Today's Lecture

- Search: motivation

⇨ Simple Search Agent and Examples

- General Search Procedure

# Search

- Often we are given a specification of what a solution is, but do not know how we would go about finding one

  we have to search for a solution

  – Enumerate a set of potential partial solutions

  – Check to see if they are solutions or could lead to one
  – We can recognize a solution once we see one

# Simple Deterministic Search Agent

Goal-driven deterministic agent with

- Perfect knowledge of the world
  - Environment changes only when the agent acts, and in known ways

- Deterministic actions
  - Agent perfectly knows:
    - actions that 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

- Agent is in a start state
- Agent is given a goal (subset of states)

- A sequence of actions taking the agent
  from the start state to a goal state is a solution

# Our simple deterministic search agent has …

perfect knowledge of its environment

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

Both of the above

None of the above

# Definition of a search problem

- Initial state
- Set of actions (operators) available to the agent: for each state, define the successor state for each operator
  - i.e., which state the agent would end up in
- Goal states
- State space: set of all possible states
  - Not necessarily given explicitly (state space might be infinite)
- Path Cost (we ignore this for now)

# Three examples

1. The delivery robot planning the route it will take in a bldg. to get from one room to another (Section 1.6)

2. Vacuum cleaner world

3. Solving an 8-puzzle

# Example 1: Delivery Robot (ch.1.6)

## Simplified

- Consider only bold locations

- Limits in direction of movement (can move only in the direction doors open)

- Start: o103
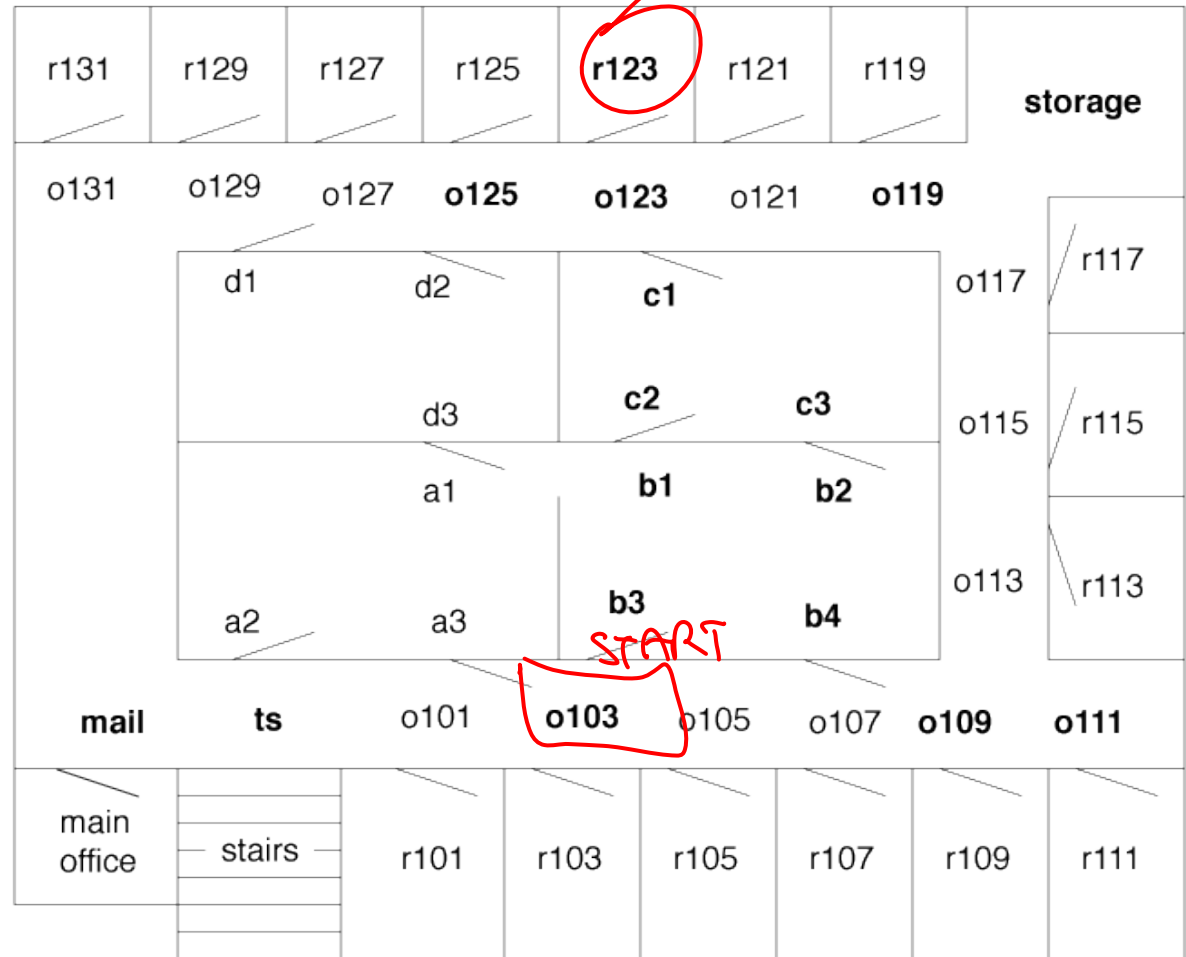
- Goal: r123
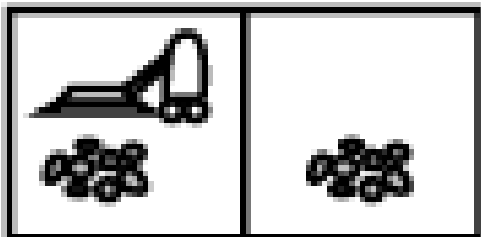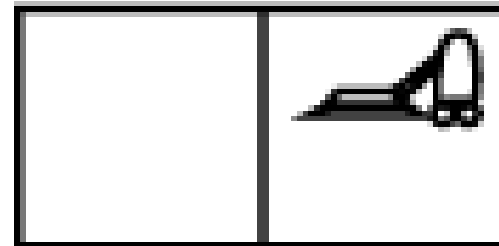
# Search Space for the Delivery Robot



## Simplified

- Consider only bold locations
- Limits in direction of movement (can move only in the direction doors open)
- Start: o103
- Goal: r123
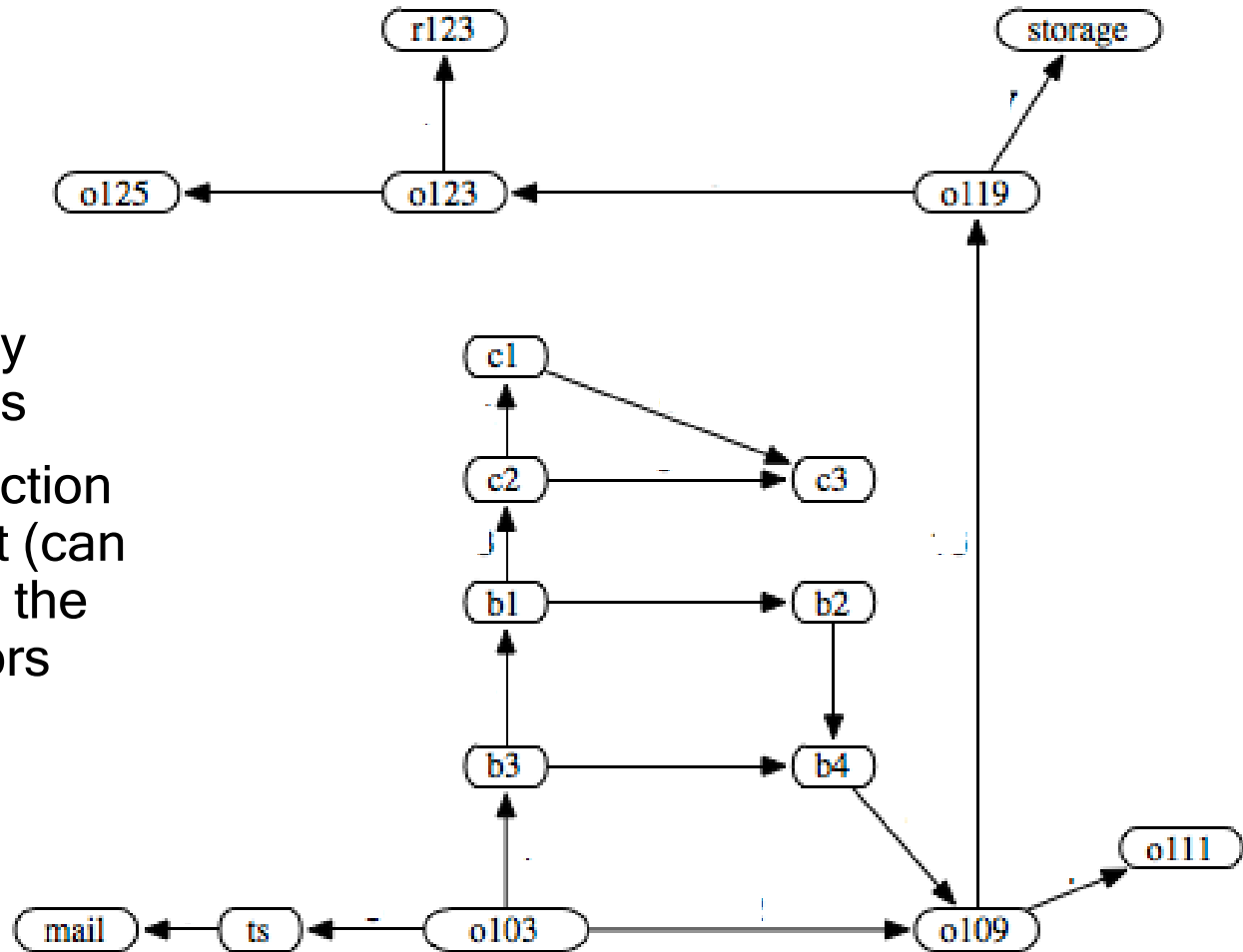
# Example 2: vacuum world

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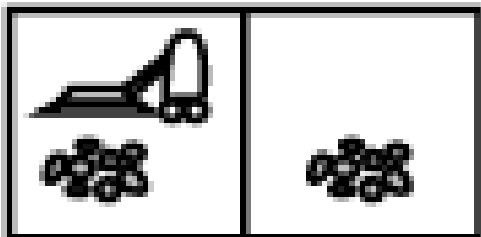  - Vacuuming agent can be in either in r1 or r2



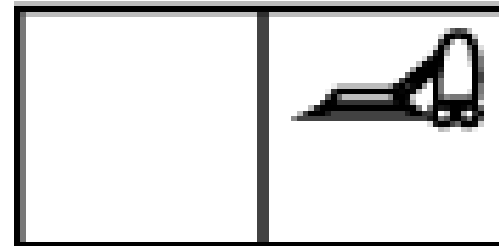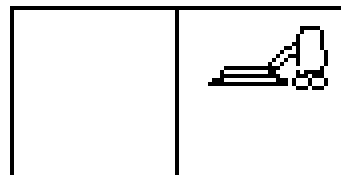Possible start state



Goal state

# Example 2: vacuum world

**Feature-based representation: how many states?**

- **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

Goal state

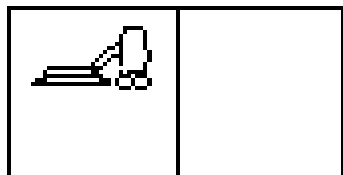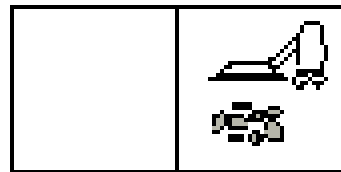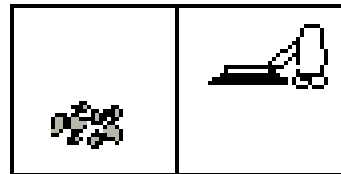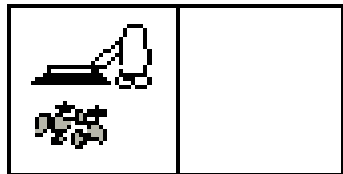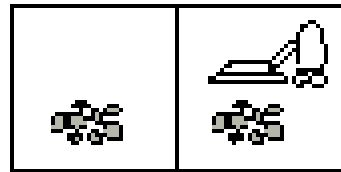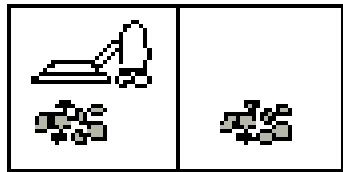# Example 2: vacuum world
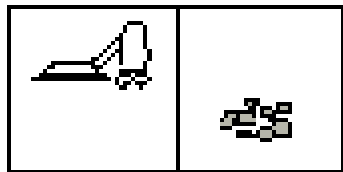


- **States** – one of the eight states in the picture
- **Operators** –*left, right, suck*
- **Possible Goal** – no dirt

# Suppose we have k rooms instead

- **The number of states is then…**

$k^3$
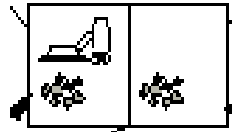
$k * 2k$

$k * 2^k$

$2 * k^k$

# Search Space



- Operators – left, right, suck
    - Successor states in the graph describe the effect of each action applied to a given state

- Possible Goal – no dirt

# Search Space



- Operators – left, right, suck
  - Successor states in the graph describe the effect of each action applied to a given state

- Possible Goal – no dirt

# Search Space



- Operators – left, right, suck
  - Successor states in the graph describe the effect of each action applied to a given state
- Possible Goal – no dirt

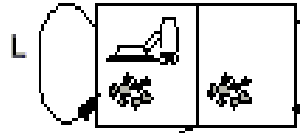# Search Space
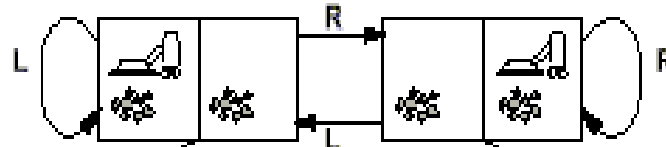


- Operators – left, right, suck
  - Successor states in the graph describe the effect of each action applied to a given state
- Possible Goal – no dirt

# Search Space
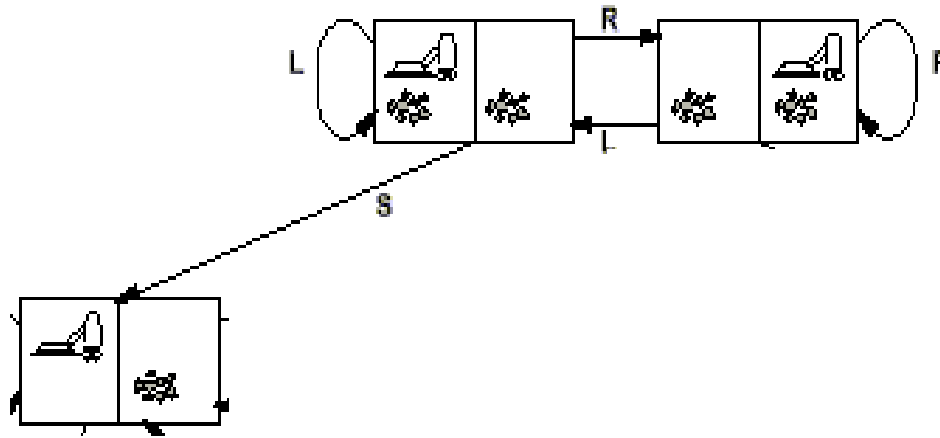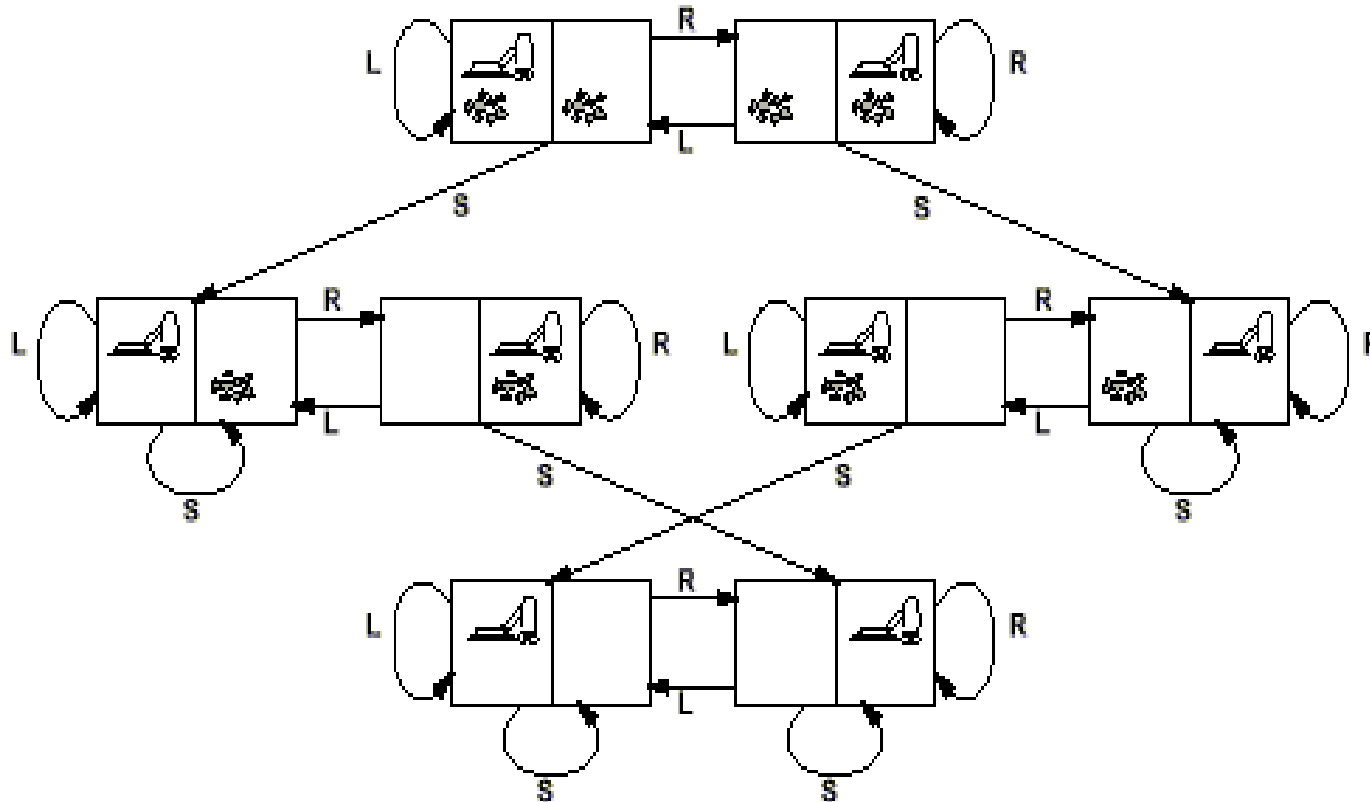


- Operators – left, right, suck
  - Successor states in the graph describe the effect of each action applied to a given state
- Possible Goal – no dirt

# Example 3: Eight Puzzle



| 5 | 4 |   |
|---|---|---|
| 6 | 1 | 8 |
| 7 | 3 | 2 |

| 1 | 2 | 3 |
|---|---|---|
| 8 |   | 4 |
| 7 | 6 | 5 |

**What describes a state?**

**What are the operators?**

# Search space for 8puzzle

# Search space for 8puzzle

# Example 3: Eight Puzzle



**Start State**



**Goal State**

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

- How many states are there?  $8^9$  $2^9$  $9^9$  $9!$

**Operators**: the blank spot moves left, right, up down

**Goal**: the configuration with all numbers in the right sequence

# Today's Lecture

- Search: motivation

- Simple Search Agent and Examples
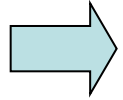
General Search Procedure

# Search: Abstract Definition

How to search

- Start at the start state

- Evaluate where actions can lead us 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 to review the formal definition of a graph...

# Graphs

- A directed graph consists of a set N of nodes (vertices) and a set A of ordered pairs of nodes, called edges (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 $n_0, n_1, .., n_k$ such that $\langle n_{i-1}, n_i \rangle \in A$.

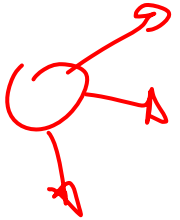- A cycle is a non-empty path such that the start node is the same as the end node.

Search graph

- Nodes are search states

- Edges correspond to actions

- Given a set of start nodes and goal nodes, a solution is a path from a start node to a goal node.

# 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
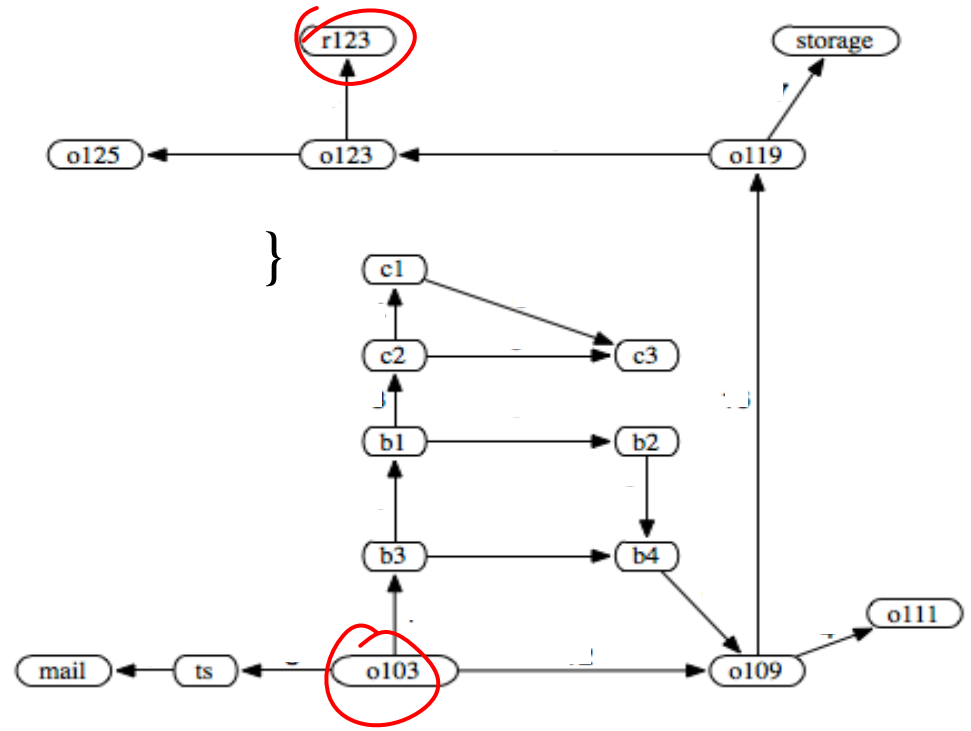
- If the forward branching factor of a node is b and the graph is a tree, there are $b^n$ nodes that are n steps away from a node

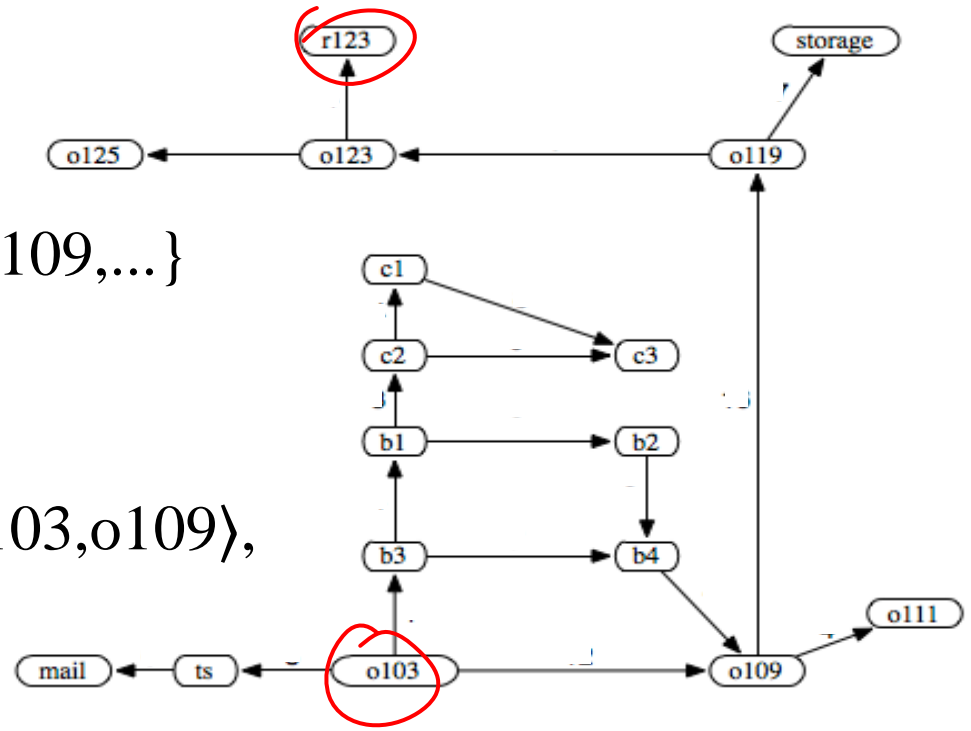Nodes={                              }

Edges={

   ...}

Solution paths:

# Graph Specification for the Delivery Robot



Nodes={mail, ts, o103, b3, o109,...}

Edges={
⟨ts,mail⟩,
⟨o103,ts⟩, ⟨o103,b3⟩, ⟨o103,o109⟩,
...}

One of three solution paths:

⟨o103, o109, o119, o123, r123⟩

# Graph Searching

- Generic search algorithm: given a graph, start nodes, and goal nodes, incrementally explore paths from the start nodes

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

- As search proceeds, the frontier expands into the unexplored nodes until a goal node is encountered.

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

# Problem Solving by Graph Searching
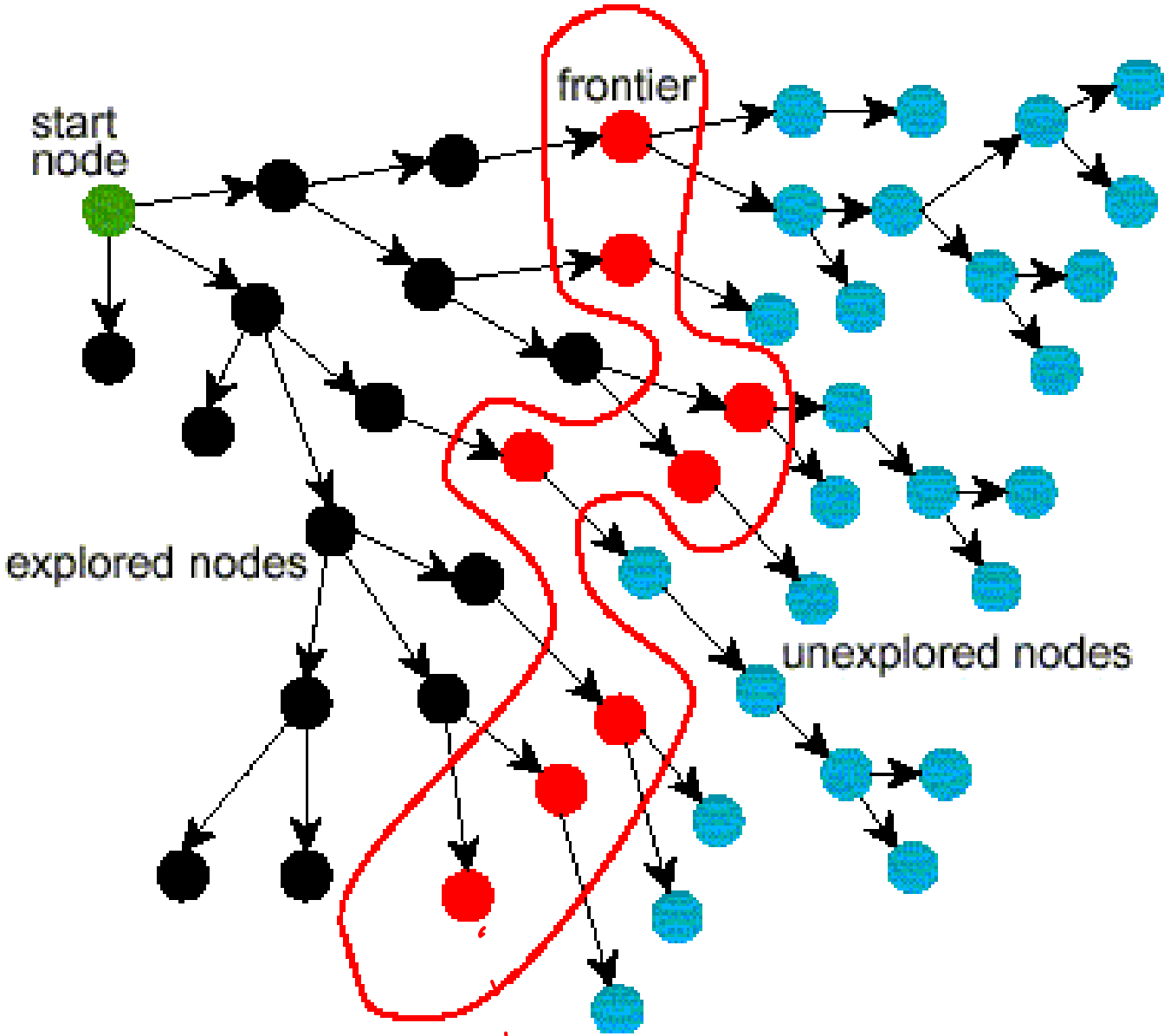
# Generic Search Algorithm

**Input:** a graph

a set of start nodes

Boolean procedure goal(n) testing 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** $<n_o,....,n_k>$;

**Else**

**For every** neighbor n of $n_k$,

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

**end**

# Lecture Summary

- Search is a key computational mechanism in many AI agents

- We will study the basic principles of search on the simple deterministic goal-driven search agent model

- Generic search approach:
  - Define a search space graph,
  - Initialize the frontier with an empty path
  - incrementally expand frontier 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 search agents
- **Assess** the size of the search space of a given search problem
- **Implement** the generic solution to a search problem

---

Coming up:

- Uninformed search: read section 3.5