Search: Intro

Computer Science cpsc322, Lecture 4

(Textbook Chpt 3.0–3.4)

May 18, 2017
Announcements

• Still looking for rooms for some TAs office hours (stay tuned)
• Straw Poll for break length
  A 15min
  B 20min
  C 25min
• Assignment 1 will be out by Tue (on Search)
People

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

Teaching Assistants
Dylan Dong wdong@cs.ubc.ca [only marking]

Johnson, David davewj@cs.ubc.ca
Office hour: ICCS TBD, Wed 1–230pm

Johnson, Jordon jordon@cs.ubc.ca
Office hour: ICCS TBD, Mon 11–1pm
TAs (cont’)

Kazemi, Seyed Mehran  smkazemi@cs.ubc.ca
Office hour: ICCS TBD, Wed 230–4pm

Rahman, MD Abed  abed90@cs.ubc.ca  Office hour: ICCS X141, Fri 3–430pm

Wang, Wenyi  wenyi.wang@alumni.ubc.ca
Office hour: TBD, mon 1–230pm
Modules we’ll cover in this course: R&Rsys

Problem

Static

Constraint Satisfaction

Query

Sequential

Planning

Representation

Reasoning Technique

First part of the course

Environment

Deterministic

Stochastic

Arc Consistency

Search

Vars + Constraints

Search

Belief Nets

Var. Elimination

Decision Nets

Var. Elimination

Markov Processes

Value Iteration

Logics

STRAIPS

Search

Factors
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
Example 1: Delivery Robot
States: each state specifies which number/blank occupies each of the 9 tiles

HOW MANY STATES? \[8^9 \times 2^9 \times 9^9 \times 9!\]

Actions: blank moves left, right, up down

Possible Goal: configuration with numbers in right sequence
Example 2: 8-Puzzle?

Possible start state

Goal state

# of states
9!

\( \approx 360 \times 10^3 \)
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

\begin{align*}
\text{loc} & \sim 2, \text{values} \{r_1, r_2\} \\
& \{r_1, \text{clean} \} \sim \{r_2, \text{unclean}\} \\
\text{room1} & \rightarrow \text{clean} \sim \text{unclean} \\
\text{room2} & \rightarrow \text{clean} \sim \text{unclean}
\end{align*}

Possible start state

Goal state

\# of states 2

2 2

\text{can be subset of states}
Suppose we have the same problem with $k$ rooms. The number of states is $\cdots$.

\[ k^3 \]
\[ k \times 2k \]
\[ k \times 2^k \]
\[ 2 \times k^k \]
Suppose we have the same problem with $k$ rooms. The number of states is $k \times 2^k$. 
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

A tiny subset!
Vacuum world: Search space graph

- States? Where it is dirty and robot location
- Actions? Left, Right, Suck
- Possible goal test? No dirt at all locations

The whole space
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...
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.

\[ N = \{ a, b, c \} \]
\[ A = \{ (a, b), (a, c) \} \]
Examples of solution

- Start state b4, goal r13
- Solution \langle b4, o107, o109, o13, r13 \rangle

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 \( n_o \), Boolean procedure \( \text{goal}(n) \) that tests if \( n \) is a goal node

**frontier:= [\langle s \rangle: s \text{ is a start node}]**;

**While** \( \text{frontier} \) is not empty:

- **select and remove** path \( \langle n_o, \ldots, n_k \rangle \) from \( \text{frontier} \);
- **If** \( \text{goal}(n_k) \)
  - **return** \( \langle n_o, \ldots, n_k \rangle \);
- **For every** neighbor \( n \) of \( n_k \)
  - **add** \( \langle n_o, \ldots, n_k, n \rangle \) to \( \text{frontier} \);

**End**
Problem Solving by Graph Searching

- Start node
- Ends of frontier
- Explored nodes
- Paths
- Unexplored nodes
- Paths
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?
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

• Uninformed search strategies
  (read textbook Sec. 3.5)

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