Recap: Achieving Goals by Search

Intelligent agents sometimes act by reflex, and sometimes they act by looking up the right answer. But these behaviors work only in limited environments.

How can an agent choose an action in a complex environment, trying to achieve a goal that it knows little about? One approach is to **search** for an answer. An agent can consider what effect each possible action would have on the world and which action—or sequence of actions—leads to the goal.

We might expect an intelligent agent to consider its options in a **systematic** way, in a way that makes some sense, either logically or based on its knowledge of the world.

For now, let's consider agents that act logically, but with relatively little knowledge about the the goal it is trying to achieve.

An Algorithm for Systematic Search

INPUT: the starting situation (the **start state**)
a goal to achieve
a search strategy

OUTPUT: a sequence of actions (called **operators**) that transforms the problem's initial state into a goal state OR an announcement that no such sequence can be found

STEPS

1. Initialize the set of states to be considered to include the start state.

2. Repeat the following:

   a. If the set of unconsidered states is empty, then announce failure.

   b. *Choose a state* to consider, based on the search strategy.

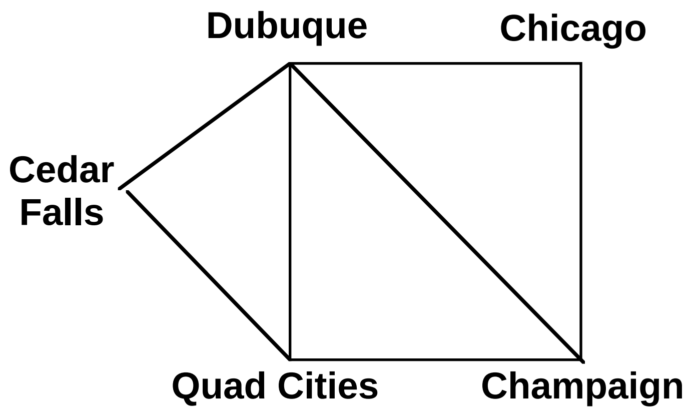
   c. If the state chosen is the goal state, then return the sequence of actions that leads to this state.

   d. Add to the set of unconsidered states all of the states that can be reached from the current state by doing one action.

Doing Systematic Search

In order to apply this algorithm, we must define the problem in the vocabulary of search.

Consider the task I face next week, when I go to my conference. I must travel from Cedar Falls to the Champaign, Illinois, area. I have decided to drive, so I have to negotiate this simplified map:



What route should I follow?

In terms of search...

- What is the initial state? Cedar Falls
- What is the goal state? Champaign
- What is the set of operators?

  The set of roads available to me:
  { Cedar Falls-Dubuque, Dubuque-Cedar Falls, ... }

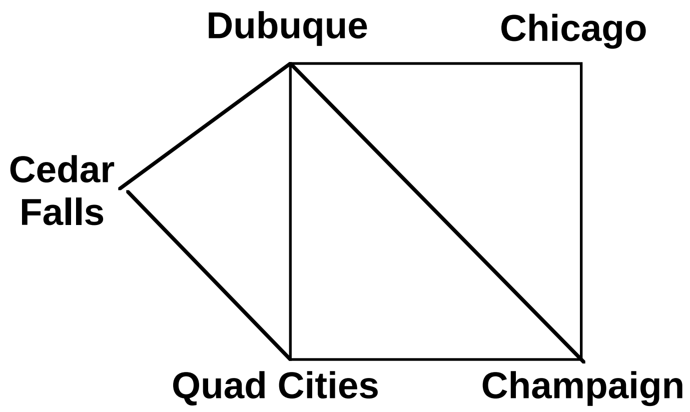
Random Systematic Search

INPUT: the start state
a goal to achieve
a search strategy

OUTPUT: a sequence of operators that transforms the problem's initial state into a goal state OR an announcement that no such sequence can be found

STEPS

1. Initialize the set of states to be considered to include the start state.
2. Repeat the following:
    a. If the set of unconsidered states is empty, then announce failure.
    b. *Choose a state to consider, based on the search strategy*.
    c. If the state chosen is a goal state, then return the sequence of actions that leads to it.
    d. Add to the set of states to explore all states that can be reached from the current one.



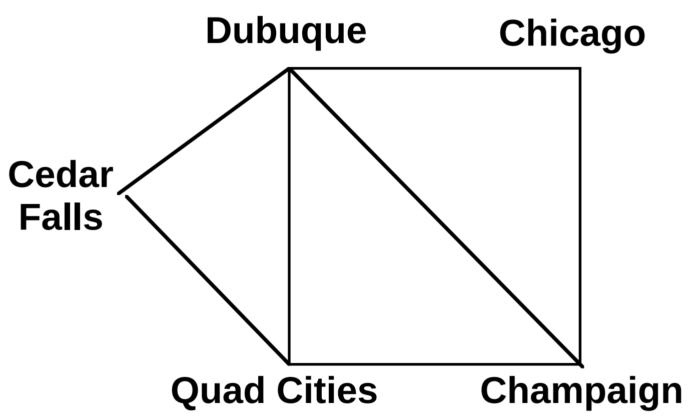
Systematic Search that Favors Old States

INPUT:
the start state
a goal to achieve
a search strategy

OUTPUT:
a sequence of operators that transforms the problem's initial state into a goal state OR
an announcement that no such sequence can be found

STEPS

1. Initialize the set of states to be considered to include the start state.
2. Repeat the following:
    a. If the set of unconsidered states is empty, then announce failure.
    b. *Choose a state to consider, based on the search strategy.*
    c. If the state chosen is a goal state, then return the sequence of actions that leads to it.
    d. Add to the set of states to explore all states that can be reached from the current one.



Systematic Search that Favors New States

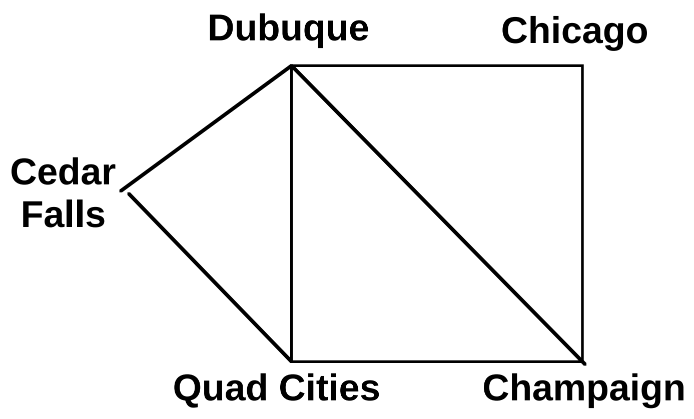
INPUT: the start state
a goal to achieve
a search strategy

OUTPUT: a sequence of operators that transforms the problem's initial state into a goal state OR an announcement that no such sequence can be found

STEPS

1. Initialize the set of states to be considered to include the start state.
2. Repeat the following:
    a. If the set of unconsidered states is empty, then announce failure.
    b. *Choose a state to consider, based on the search strategy.*
    c. If the state chosen is a goal state, then return the sequence of actions that leads to it.
    d. Add to the set of states to explore all states that can be reached from the current one.

.



Evaluating Search Strategies

With multiple search strategies available to us, we have to be able to determine what their strengths and weaknesses are objectively. That way, when faced with a particular problem, we—or our program!—will be able to make a good choice of search strategy to use.

We can evaluate a search strategy using several criteria:

- completeness

  Does it guarantee to find a solution if one exists?

- time

  How long does it take?

- space

  How much memory is required?

- optimality

  Does it find the “best” solution, if there are many?

Evaluating Search Strategies

A strategy with a bias toward old states is called **breadth-first**. How does breadth-first search (BFS) measure up?

• complete?	yes
• time?	generally expensive
• space?	generally expensive
• optimal?	only if all operators “cost” the same

The expensiveness of BFS is a product of the problem’s *branching factor*. If there aren’t many operators available in a given state, then the cost of BFS can be low. But...

A strategy with a bias toward old states is called **depth-first**. How does depth-first search (DFS) measure up?

• complete?	only if the search space is finite
• time?	modest
• space?	modest
• optimal?	no!

The low cost of space in DFS is a result of only needing to store the path leading to a state, plus immediate siblings of each state on the path. What about time?

Search in Context

Let's step back for a second from the details of search to remind ourselves of what we've done. In order to plan ahead to find a solution, an intelligent agent goes through three distinct phases:

1. Formulate the goal.

   What is desired?

   For elective goals, this is not trivial. And it is something that our programs don't often worry about, because the goals are specified by the programmer!

2. Formulate the problem.

   What needs to be done?

   If an agent plans to search for a path to its goal, then this is the step where it must define the problem in the vocabulary of search: how to represent a state, what the initial state is, what the operators are, and what the goal test is.

Search in Context

The result of "problem formulation" is a concrete definition of the problem:

- the **state space**
    - What states will be considered?
    - What actions will be considered?
    - What is the initial state?

- the **goal test**
  How does the agent know when it is done?

- (later) a **path cost function**
  How will we compare potential solutions?

Finally...

3. Formulate the solution.

   What is the answer?

   If an agent plans to search for a path to its goal, then this is the step where it applies its search algorithm to search through (potentially, all possible) states for a path to a goal state. At this step, it must choose a search bias.

Formulating the Problem

Formulating a problem for search requires:

1. defining what states look like

2. defining what operations are possible and relevant, and how each affects the state of the world

3. defining how to tell when the search is done

This often isn’t as easy as it seems...

1. In some environments, the agent can determine which state it is in, and it knows exactly the effect of each action. These are the easy problems.

2. In some environments, the agent either cannot tell which state it is in or does not know exactly the effect of each action. These are harder!

3. In some environments, the agent is ignorant both of its initial state and of the effects of its actions. These are hardest!!

Exploring the World

The last sort of problem is called **exploration**.

The agent must sense during execution and try to learn the effects of its actions and a way to characterize the state of the world.

The reasoning component of the agent cannot think ahead with enough precision to map out one sequence of actions to solve the problem.

In such situations, an agent tends to **interleave** its thinking and its acting. It needs to begin acting before it finishes reasoning, in order to use the information gained during the acting step.

Formulating a Solution

This last step is where the agent actually searches for something.

Search involves using an algorithm like the one we’ve been considering to explore the set of possible states, in search of a sequence of operations that leads from the initial state to a goal state.

Because search employs an algorithm, formulating a solution is **objective:** the algorithm defines the exact order in which states will be explored.

This is different from problem formulation, which is **subjective**: there are multiple formulations for any problem that satisfy the definitions of our terms.

Uninformed Search

Random, breadth-first, and depth-first search strategies are considered **uninformed** because they do not use any knowledge about the problem being solved. The algorithm we've been considering works as well for finding a route between two cities as it does for solving a crossword puzzle, as well for scheduling experiments on the Hubble Space Telescope as it does for proving that a logical statement is true.

This generality is both a source of strength and of weakness:

An intelligent agent should be able to solve problems that it hasn't seen before, by using a general intelligence about how to solve problems. So uninformed search is an essential tool for an intelligent agent.

But algorithms that work *everywhere* don't tend to work very well *anywhere*, which means that they will be sub-optimal in some important way for almost every problem.

How can an agent use knowledge of the world to do a **better** job solving problems?