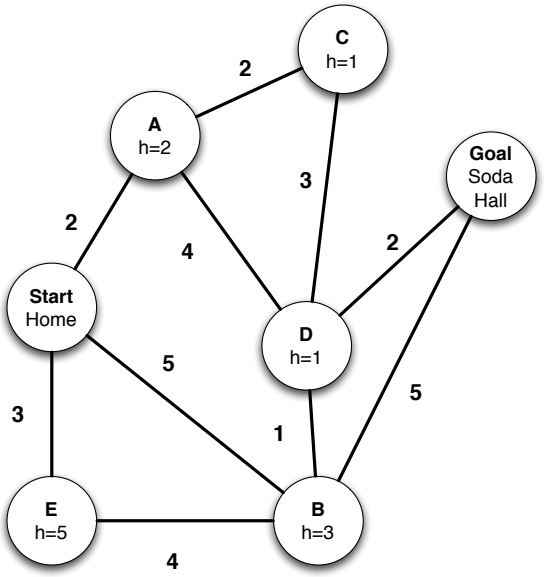
CS 188 Fall 2007 Introduction to Artificial Intelligence Section Handout #1

Problems to do in section

1. Uniform Cost vs. A* Search



Our intrepid hero, Search Agent, is late for her artificial intelligence class and needs to get there fast! The graph above represents how long it takes Search Agent to walk between different parts of campus.

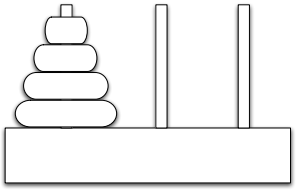
a) Use uniform-cost search to help Search Agent get from Home to Soda Hall. Keep track of the nodes on the priority queue for each step of the algorithm. Assume ties are broken alphabetically.

b) Now imagine Search Agent shows her graph to some friends and asks them to give her an estimate of how long it takes them to walk from that spot to Soda hall. Search Agent writes this information as heuristic guesses (h values) in each of the states. Use A* to help Search Agent get from Home to Soda Hall. Keep track of the nodes on the priority queue for each step of the algorithm. Assume ties are broken alphabetically.

c) In general, what do we have to know about Search Agents friends' estimates in order to be sure that A* is giving us a lowest cost solution?

2. Designing an A* Heuristic for Towers of Hanoi

The Towers of Hanoi is a famous problem for studying recursion in computer science and recurrence equations in discrete mathematics. There are N discs of varying sizes on a peg, and two empty pegs. We are allowed to move a disc from one peg to another as long as we never move a larger disc on top of a smaller disc. The goal is to move



all the discs to the rightmost peg (see figure above). There is a very interesting story regarding the origin of the problem:

It is said that after creating the world God set on Earth three rods made of diamond and 64 rings of gold. At the creation they were threaded on one of the rods in order of size, the largest at the bottom and the smal lest at the top. God also created a monastery close to the rods. The monks task in life is to transfer al l the rings onto another rod. The only operation permitted consists of moving a single ring from one rod to an- other, in such a way that no ring is ever placed on top of another smaller one. When the monks have finished their task, according to the legend, the world will come to an end.

In this problem well formulate the problem as a search problem and develop a heuristic for solving it.[1]

a) Formalize the Towers of Hanoi as a search problem :

- What is the search state
- At a given state, what are the actions we can take and what are the resulting states
- What is the cost of an action
- What is the goal test

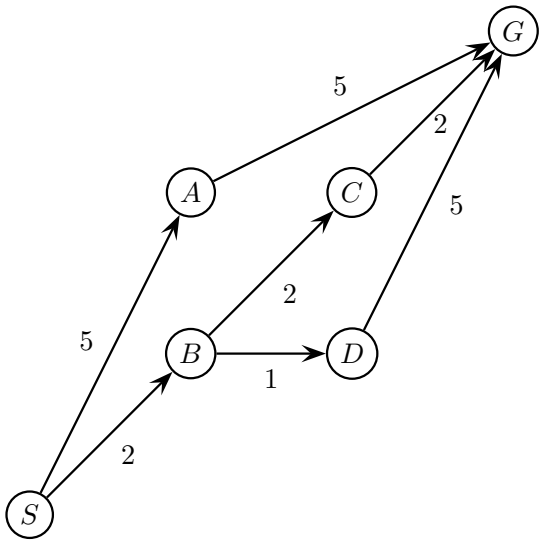
b) Develop a heuristic for the Towers of Hanoi. From a given state, can you tell at least how many actions it will take to reach the goal state? Is your heuristic admissible? Why?

[1]The CS 188 staff in no way endorses the hastening of the apocalypse through algorithmic or other means.

Problems to do on your DUE: Beginning of section by 9/12/07

1. Search

Consider the following search problem:



Node	h
S	4
A	3
B	2
C	100
G	0
D	0

Which path will each search algorithm return, assuming all successor functions work out in such a way that nodes are explored in alphabetical order whenever possible?

(a) Breadth-first search

(b) Depth-first search

(c) Uniform-cost search

(d) A* search

(e) Greedy search

(f) Name a node that uniform-cost search will expand, but A* will not.

Describe a reasonably general case in which each of the following will occur, or state that the scenario is impossible. Correct answers should not take more than one or two sentences.

(g) Depth-first search never terminates, despite a finite goal

(h) Breadth-first search never terminates, despite a finite goal

(i) Uniform cost search and breadth-first search expand the same nodes and return the same goal

2. Burrito Search Problem

You manage a taqueria. Making a burrito requires a set of tasks $T_1, \ldots, T_n$. Each task T_i takes t_i minutes to complete. Also, some tasks require other tasks to be completed before they can begin. You may schedule tasks among your employees. How fast can you make a burrito?

Concrete example: You have two ($e = 2$) employees and the following tasks, with requirements and times listed below:

- steam tortilla, 1
- cook steak, 3
- add beans (steam), 1
- add cheese (steam), 1
- add steak (steam, cook steak), 1
- wrap burrito (add all), 1
- prepare to-go bag, 3
- collect money from customer, 3
- put burrito in a prepared bag (wrap, prepare), 1

a) Formulate the problem of preparing a burrito as a search problem where you have an arbitrary number of employees e. In particular:

- What is the state space?
- What actions are available in each search state and what are the costs associated with an action?
- What is the initial search state?
- How do we detect a goal state?

b) Propose an admissible heuristic for this problem. Explain why your heuristic is admissible?