# Heuristic Search

Introduction to
Artificial Intelligence
COS302
Michael L. Littman
Fall 2001

# Administration

Thanks for emails to Littman@cs!

Test message tonight… let me know if you don't get it.

Forgot to mention reading… (sorry).

# AI in the News

"Computers double their performance every 18 months. So the danger is real that they could develop intelligence and take over the world." *Eminent physicist* Stephen Hawking*, advocating genetic engineering as a way to stay ahead of artificial intelligence.* [Newsweek, 9/17/01]

# Blind Search

Last time we discussed BFS and DFS and talked a bit about how to choose the right algorithm for a given search problem.

But, if we know about the problem we are solving, we can be even cleverer…

# Revised Template

- fringe = {($s_0$, f($s_0$)}; /* initial cost */
- markvisited($s_0$);
- While (1) {

```
  If empty(fringe), return failure;
  (s, c) = removemincost(fringe);
  If G(s) return s;
  Foreach s' in N(s)
    if s' in fringe, reduce cost if f(s') smaller;
    else if unvisited(s') fringe U= {(s', f(s'))};
    markvisited(s');
}
```

# Cost as True Distance

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| | | | | 2 | 1 | 0 | |
| | | | 3 | 2 | 1 | 1 | |
| | 5 | 4 | 3 | 2 | 2 | 2 | |
| | 5 | 4 | 3 | 3 | 3 | | |
| | 5 | 4 | 4 | 4 | | | |
| | | | | | | | |
| | | | | | | | |

## Some Notation

**Minimum (true) distance to goal**
- **t(s)**

**Estimated cost during search**
- **f(s)**

**Steps from start state**
- **g(s)**

**Estimated distance to goal (heuristic)**
- **h(s)**

## Compare to Optimal

**Recall b is branching factor, d is depth of goal ($d=t(s_0)$)**

**Using true distances as costs in the search algorithm (f(s)=t(s)), how long is the path discovered?**

**How many states get visited during search?**

## Greedy

**True distance would be ideal. Hard to achieve.**

**What if we use some function h(s) that *approximates* t(s)?**

**f(s) = h(s): expand closest node first.**

## Approximate Distances

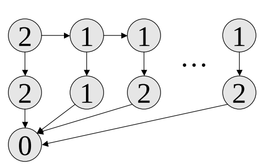
**We saw already that if the approximation is perfect, the search is perfect.**

**What if costs are +/- 1 of the true distance?**

$$|h(s)-t(s)| \leq 1$$

## Problem with Greedy

**Four criteria?**




## Algorithm A

**Discourage wandering off:**

**f(s) = g(s)+h(s)**

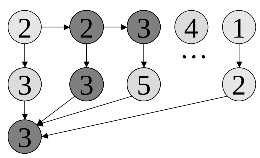
**In words:**

**Estimate of total path cost: cost so far plus estimated completion cost**

**Search along the most promising path (not node)**

## A Behaves Better

Only wanders a little




## A Theorem

If h(s) ≤ t(s) + k (overestimate bound), then the path found by A is no more than k longer than optimal.

## A Proof

f(goal) is length of found path.

All nodes on optimal path have

f(s) = g(s) + h(s)
≤ g(s) + t(s) + k
= optimal path + k

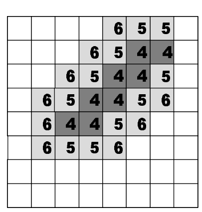
## How Insure Optimality?

Let k=0! That is, heuristic h(s) must always underestimate the distance to the goal (be optimistic).

Such an h(s) is called an "admissible heuristic".

A with an admissible heuristic is known as A*. (Trumpets sound!)

## A* Example



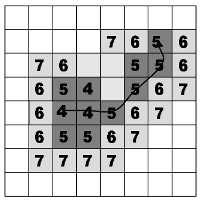

## Time Complexity

Optimally efficient for a given heuristic function: No other complete search algorithm would expand fewer nodes.

Even perfect evaluation function could be $O(b^d)$, though. When?

Still, more accurate better!

## Simple Maze




## Relaxation in Maze

Move from (x,y) to (x',y') is illegal

- ~~If |x-x'| > 1~~
- ~~Or |y-y'| > 1~~
- ~~Or (x',y') contains a wall~~

Otherwise, it's legal.

## Relaxations Admissible

Why does this work?

Any legal path in the full problem is still legal in the relaxation. Therefore, the optimal solution to the relaxation must be no longer than the optimal solution to the full problem.

## Relaxation in 8-puzzle

Tile move from (x,y) to (x',y') is illegal

- If ~~|x-x'| > 1 or |y-y'| > 1~~
- Or (|x-x'| ≠ 0 and |y-y'| ≠ 0)
- ~~Or (x',y') contains a tile~~

Otherwise, it's legal.

## Two 8-puzzle Heuristics

- $h_1(s)$: total tiles out of place
- $h_2(s)$: total Manhattan distance

Note: $h_1(s) \leq h_2(s)$, so the latter leads to more efficient search

Easy to compute and provides useful guidance

## Knapsack Example

Optimize value, budget: $10B.

- Mark. cost value
- NY 6 8
- LA 5 8
- Dallas 3 5
- Atl 3 5
- Bos 3 4

## Knapsack Heuristic

- State: Which markets to include, exclude (some undecided).
- Heuristic: Consider including markets in order of value/cost. If cost goes over budget, compute value of "fractional" purchase.
- Fractional relaxation.

## Memory Bounded

- Just as iterative deepening gives a more memory efficient version of BFS, can define IDA* as a more memory efficient version of A*.
- Just use DFS with a cutoff on f values. Repeat with larger cutoff until solution found.

## What to Learn

- The A* algorithm: its definition and behavior (finds optimal).
- How to create admissible heuristics via relaxation.

## Homework 2 (partial)

1. Consider the heuristic for Rush Hour of counting the cars blocking the ice cream truck and adding one. (a) Show this is a relaxation by giving conditions for an illegal move and showing what was eliminated. (b) For the board on the next page, show an optimal sequence of boards *en route* to the goal. Label each board with the f value from the heuristic.
2. Describe an improved heuristic for Rush Hour. (a) Explain why it is admissible. (b) Is it a relaxation? (c) Label the boards from 1b with the f values from your heuristic.

## Rush Hour Example