

AIMA Chapter 4, Lecture 1, 20 September 2006

A. Return exams and exam post-mortem. Assign homework

Read Ch. 4.1 and 4.2. Prepare orally Question #13 from Day 1 handout and answer:

Are these admissible?

In 8-puzzle, $h_2(n)$ = number of tiles out of place

In missionaries & cannibals $h_3(n)$ = number of m&c on other bank

B. Review: We said that among blind (uninformed) searches,

Iterative deepening is complete (always gives a solution eventually) & optimal (correct).

Uniform cost search (expand node which gives lowest path cost $g(n)$)

is complete and optimal **if** the cost at each node is at least larger than $\epsilon > 0$ (same ϵ for all nodes), but is usually worse in time than iterative deepening.

C. How did you fare on the homework problems?

What did you do with #9 re Richard Nixon? [Point was to get students to admit that logic isn't everything, or at least that in speaking English we often are imprecise.]

What did you do with the monkeys & bananas? [Point was to get students to eventually bound a problem by saying "and nothing else is going to matter." And to think about a representation language.]

- 1 monkey is hungry
- 2 if monkey is hungry and monkey has banana, monkey eats banana
- 3 if monkey eats banana, monkey is not hungry [note time sequence matters for 1, 3]
- 4 if monkey on floor, monkey climbs on box [note dead end if box not under banana]
- 5 if monkey not on box, monkey moves box under bananas
- 6 if monkey on box and box under banana, monkey can reach banana [given]
- 7 if monkey can reach banana, monkey has banana

This is almost Prolog. No *not* is allowed in the head of a clause in Prolog.

- 1 hungry(monkey, 0).
- 2 eats(monkey, banana, Time) :- hungry(monkey, Time), has(monkey, banana).
- 3 not hungry(monkey, Time2) :- eats(monkey, banana, Time1), Time2 > Time1.
- 4 on(monkey, box, Time2) :- on(monkey, floor, Time1), Time2 > Time1.
- 5 under(box, banana) :- not on(monkey, box). [moving implicit]
- 6 can_reach(monkey, banana) :- on(monkey, box, time), under(box, banana).
- 7 has(monkey, banana) :- can_reach(monkey, banana).

We might be able to get around the limitation by creating a new predicate to use in 3:

- 8 nothungry(monkey, Time2) :- eats(monkey, banana, Time1), Time2 > Time1.

But then we'd have to add some rule to relate hungry and not hungry:

- 9 nothungry(monkey, Time) :- not hungry(monkey, Time).

D. Textbook monkeys & bananas problem, p. 90, 3.7 b: I put the solution on Q:, as I emailed.

E. What's in a state (first two from previous lecture)

- n, current node
- The path to current node
- d, depth of current node
- g(n), path-cost from start state to current node [standard notation]

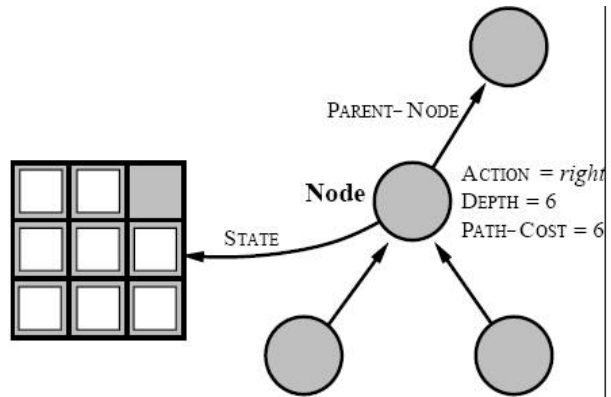


Fig. 3.8 on p. 71

F. Sections 4.1–4.2: Heuristic searches (also called informed or best-first searches)

Evaluation function $f(n)$

An estimate of the distance from the start state to the goal state when at state n .
Small values of $f(n)$ are good!

Remember that $h(n)$ is a measure of how far the goal state is from state n .

1. Greedy uses $f(n) = h(n)$
Simple, but not optimal (solution may not be correct), and not complete (may not give a solution).

2. A* uses $f(n) = g(n) + h(n)$

E.g. $h_1(n)$ = straight-line distance in a map.

A* is optimal and complete **if** $h(n)$ **admissible**:
always underestimates the cost to reach the goal
(p. 97)

$h_1(n)$ is admissible for traveling by map.

| | | | |
|-----------|-----|----------------|-----|
| Arad | 366 | Mehadia | 241 |
| Bucharest | 0 | Neamt | 234 |
| Craiova | 160 | Oradea | 380 |
| Dobreta | 242 | Pitesti | 100 |
| Eforie | 161 | Rimnicu Vilcea | 193 |
| Fagaras | 176 | Sibiu | 253 |
| Giurgiu | 77 | Timisoara | 329 |
| Hirsova | 151 | Urziceni | 80 |
| Iasi | 226 | Vaslui | 199 |
| Lugoj | 244 | Zerind | 374 |

Fig. 4.1 on p. 95

Otherwise: we may be lucky anyway.

Besides: A* is a space hog, so we'd need memory-bounded refinements.

3. Local methods: Remember only current state, not whole unfolding state tree.