

COMP9414: Artificial Intelligence

Lecture 2b: Informed Search

Wayne Wobcke

e-mail: w.wobcke@unsw.edu.au

Informed (Heuristic) Search

- Uninformed methods of search are capable of systematically exploring the state space in finding a goal state
- However, uninformed search methods are very inefficient
- With the aid of problem-specific knowledge, informed methods of search are more efficient
- All implemented using a **priority queue** to store frontier nodes

This Lecture

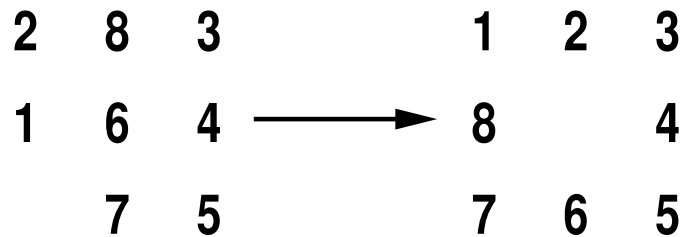
- Heuristics
- Informed Search Methods
 - ▶ Best-First Search
 - ▶ Greedy Search
 - ▶ A* Search
 - ▶ Iterative Deepening A* Search

Heuristics

- Heuristics are “rules of thumb”
- Heuristics are criteria, methods or principles for deciding which among several alternative courses of action promises to be the most effective in order to achieve some goal. “Heuristics” (Pearl 1984)
- Can make use of heuristics in deciding which is the most “promising” path to take during search
- In search, heuristic must be an underestimate of actual cost to get from current node to **any** goal — an **admissible heuristic**
- Denoted $h(n)$; $h(n) = 0$ whenever n is a goal node

Heuristics — Example

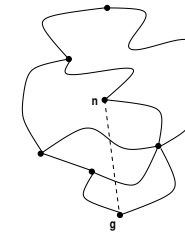
- 8-Puzzle — number of tiles out of place



- Therefore $h(n) = 5$

Heuristics — Example

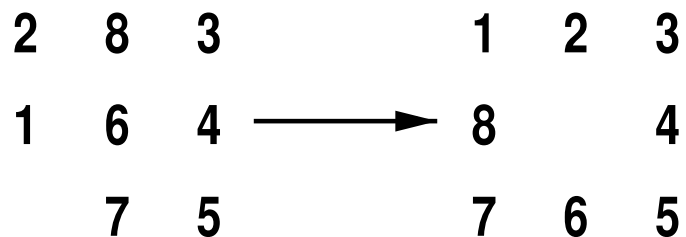
- Another common heuristic is the straight-line distance (“as the crow flies”) from node to goal



- Therefore $h(n) = \text{distance from } n \text{ to } g$

Heuristics — Example

- 8-Puzzle — **Manhattan distance** (distance tile is out of place)

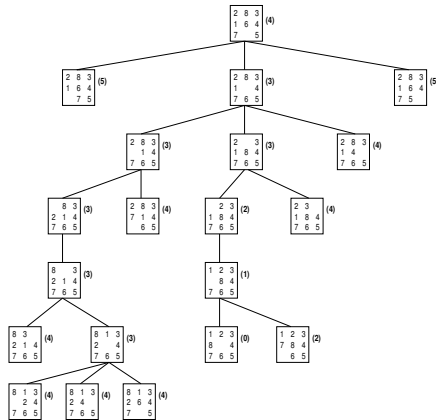


- Therefore $h(n) = 1 + 1 + 0 + 0 + 0 + 1 + 1 + 2 = 6$

Greedy Search

- Idea:** Expand node with the smallest estimated cost to reach the goal
- Use heuristic function $h(n)$ to order nodes on frontier, i.e. choose node for expansion with lowest $h(n)$
- Analysis
 - ▶ Similar to depth-first search; tends to follow single path to goal
 - ▶ Not optimal, incomplete
 - ▶ Time $O(b^m)$; Space $O(b^m)$
 - ▶ However, good heuristic can reduce time and space complexity significantly

Greedy Search



A* Algorithm

OPEN — nodes on frontier; CLOSED – expanded nodes

OPEN = $\{\langle s, nil \rangle\}$

while OPEN is not empty

remove from OPEN the pair $\langle n, p \rangle$ with minimal $f(n)$

place $\langle n, p \rangle$ on CLOSED

if n is a goal node **return** success (path p)

for each edge e connecting n and n' with cost c

if $\langle n', p' \rangle$ is on CLOSED **then if** $p \oplus e$ is cheaper than p'

then remove n' from CLOSED and put $\langle n', p \oplus e \rangle$ on OPEN

else if $\langle n', p' \rangle$ is on OPEN **then if** $p \oplus e$ is cheaper than p'

then replace p' with $p \oplus e$

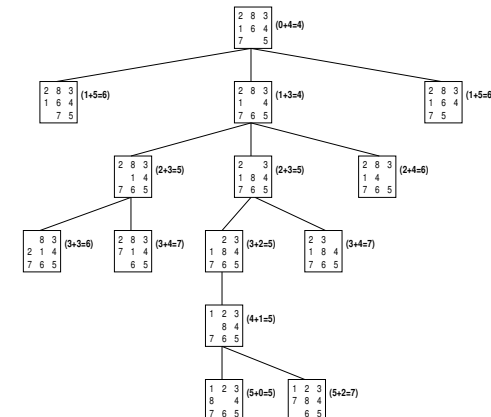
else if n' is not on OPEN **then** put $\langle n', p \oplus e \rangle$ on OPEN

return failure

A* Search

- **Idea:** Use **both** cost of path generated and estimate to goal to order nodes on the frontier
- $g(n)$ = cost of path from start to n ; $h(n)$ = estimate from n to goal
- Order priority queue using function $f(n) = g(n) + h(n)$
- $f(n)$ is the estimated cost of the cheapest solution extending this path
- Expand node from frontier with smallest f -value
- Essentially combines uniform-cost search and greedy search

A* Search



A* Search — Analysis

Subject to conditions on next slide:

- Optimal (and optimally efficient)
- Complete
- Number of nodes searched (and stored) still exponential in worst case
 - ▶ Unless the error in the heuristic grows no faster than the log of the actual path cost $h^*(n)$ of reaching a goal from n :

$$|h(n) - h^*(n)| \leq O(\log h^*(n))$$
 - ▶ Which almost never happens: for many heuristics, this error is at least proportional to the path cost

A* Search – Optimal Efficiency

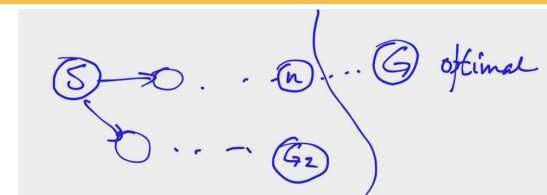
- A* is **optimally efficient** for a given heuristic: of the **optimal** search algorithms that expand search paths from the root node, there is no other optimal algorithm that expands fewer nodes in finding a solution
- **Monotonic** heuristic — along any path, the f -cost never decreases
- Common property of admissible heuristics
- If this property does not hold then the path cost for a node n' , a successor of node n , can be set to:

(Pathmax Equation) $f(n') = \max(f(n), g(n') + h(n'))$

A* Search – Optimality

- Conditions on state space graph
 - ▶ Each node has a finite number of successors
 - ▶ Every arc in the graph has cost greater than some $\epsilon > 0$
- Condition on heuristic function $h(n)$: **admissibility**
 - ▶ For every node n , the heuristic never overestimates the actual cost $h^*(n)$ of reaching a goal from n , i.e. $h(n) \leq h^*(n)$

Proof of the Optimality of the A* Algorithm



G : optimal goal node; G_2 : another goal node selected by A*
 n : node on frontier on optimal path to G ; $h^*(n)$: true cost to goal from n
 Suppose A* chose G_2 rather than n
 Then: $g(G_2) = f(G_2) \leq f(n)$ since G_2 is a goal node and A* chose G_2
 $= g(n) + h(n)$ by definition
 $\leq g(n) + h^*(n)$ by admissibility
 $\leq f(G)$ since G is on a path from n
 $= g(G)$ since G is a goal node

This means G_2 , and hence any node returned by A*, is optimal

Heuristics — Properties

- h_2 **dominates** h_1 iff $h_2(n) \geq h_1(n)$ for any node n
- A* expands fewer nodes on average using h_2 than h_1
 - ▶ Every node for which $f(n) < f^*$ is expanded
So n is expanded whenever $h(n) < f^* - g(n)$
So any node expanded using h_2 is expanded using h_1
 - ▶ Always better to use an (admissible) heuristic with **higher** values
- Suppose there are a number of admissible heuristics for a problem $h_1(n), h_2(n), \dots, h_k(n)$
 - ▶ Then $\max_{i \leq k} h_i(n)$ is a more powerful admissible heuristic
 - ▶ Therefore can design a range of heuristics for special cases

Iterative Deepening A* Search

- IDA* performs repeated depth-bounded depth-first searches as in Iterative Deepening, however the bound is based on $f(n)$
- Start by using f -value of initial state
- If search ends without finding a solution, repeat with new bound of minimum f -value exceeding previous bound
- IDA* is optimal and complete with the same provisos as A*
- Due to depth-first search, space complexity = $O(\frac{bf^*}{\delta})$ (where δ = smallest operator cost and f^* = optimal solution cost) — often $O(bd)$ is a reasonable approximation
- Another variant – SMA* (Simplified Memory-Bounded A*) – makes full use of memory to avoid expanding previously expanded nodes

Generating Heuristics

- Admissible heuristics can be derived from the **exact** solution cost of a **relaxed** version of the problem
- If the rules of the 8-puzzle are relaxed so that a tile can move **anywhere**, then #tiles-out-of-place gives the shortest solution
- If the rules are relaxed so that a tile can move to **any adjacent square**, then Manhattan distance gives the shortest solution
- For TSP: let path be **any** structure that connects all cities
 \implies minimum spanning tree heuristic

Conclusion

- Informed search makes use of problem-specific knowledge to guide progress of search
- This can lead to a significant improvement in performance
- Much research has gone into admissible heuristics
- Even on the automatic generation of admissible heuristics